Smart Usage of Context Information for the Analysis, Design, and Generation of Power-Aware Policies for Mobile Sensing Apps

Technical Report

LTI-TR-2014-07

Student Rafael Pérez-Torres

Advisors César Torres Huitzil Phd, Hiram Galeana Zapién Phd.

Information Technology Laboratory,
CINVESTAV Tamaulipas

Abstract

This technical report describes the progress and advances developed for the thesis work titled *Smart Usage of Context Information for the Analysis, Design, and Generation of Power-Aware Policies for Mobile Sensing Apps* during the first year of the doctoral program.

In summary, the different efforts during the first year have been focused on producing a solid state of the art revision, as well as on defining the core elements that will power up the methodology for solving the main problem.

Keywords. mobile sensing apps, energy consumption, context information, policy, smartphone.

1 Introduction

OK REMEMBER THIS IS TECHNICAL, YOU CAN DESCRIBE AT TECHNICAL DETAIL THE SECTIONS This technical report describes the progress and advances developed for the thesis work titled *Smart Usage of Context Information for the Anal-*

ysis, Design, and Generation of Power-Aware Policies for Mobile Sensing Apps during the first year of the doctoral program.

In summary, the different efforts during the first year have been focused on producing a solid state of the art revision, as well as on defining the core elements that will power up the methodology for solving the main problem.

In order to ease the comprehension for the reader, this report has been structured as a self-contained document, briefly addressing the scientific context of the research, describing the hypothesis, problem statement and objectives. This technical report is structured as follows. Section 2 presents a brief review of the context of the research. Section 3 is prepared as for describing the study performed on state-of-art techniques, covering a taxonomy of previous solutions and a framework for their analysis. Section C is aimed at describing the advances produced on our proposed solution, including the insights for achieving learning from user and a general overview of the mechanisms for achieving sensor usage adaptations. Section D addresses some FUTURE WORK! Section E summarizes the scientific products prepared during this first year of work. Finally, Section F present the conclusions of this report.

2 Research description

This section is aimed at giving an overall description of the context of the research, in order to ease the comprehension of the reader and producing a self-conteined document.

The popularity of smartphones is a remarkable instance of the adoption of technology by society. According to the Ericsson Mobility report [Ericsson, 2015] there were 3,400 million of smartphone subscriptions and 250 million of mobile PCs, tablets and mobile router subscriptions in the 2015. The presence of smartphones is drastically changing our daily lives by providing an increasingly powerful set of tools as a result of significant advances in the field. The capabilities and platform complexity of smartphones are continuously improving, becoming truly system-on-chip (SoC) devices able to adapt their operation over three distinguishing dimensions, namely communication, sensing, and computation, as conceptually shown in Figure 2.1. The advances on these dimensions have enabled the *context-awareness* [1] paradigm in mobile and pervasive computing, opening the path for a myriad of applications and services. Particularly in the sensing dimension, modern smartphones include a set of low cost and powerful embedded sensors such as accelerometer, digital compass, gyroscope, GPS receiver, microphone, and camera, among others, providing them with rich capabilities for cre-

[1]For the sake of clarity, we stick to the definition of *context* proposed by [Chen and Kotz, 2000], as the set of environmental states and settings that either determines the application's behavior or in which an application event occurs and is interesting to the user.

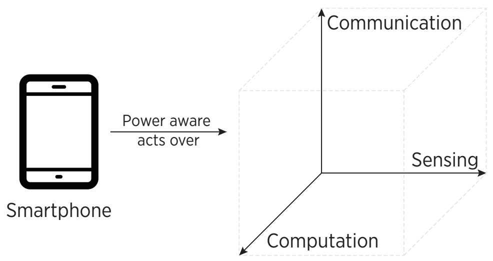

Figure 2.1: The communication, computation, and sensing features offered by smartphones represent the main dimensions where the functionality of these devices relies on.

ating mobile sensing applications at different spatial and temporal scales [Lane et al., 2010, Campbell and Choudhury, 2012, Kjaergaard, 2012] and with different levels of accuracy and energy costs. For this class of applications, the core requirements are the background and continuous sensors' data acquisition, as well as the associated on-device computations for extracting useful information for the user [Lane et al., 2010, Ra et al., 2012]. As user interaction can be required for collecting data, two sensing paradigms can be implemented [Lane et al., 2010]. The *opportunistic paradigm* tries to determine the most appropriate moment for automatically collecting data from sensors without human participation at all. On the other hand, the *participatory paradigm* leverages user's abilities requiring his/her participation to describe data and choose the moment for collecting them. Given its autonomy, the opportunistic paradigm represents the preferable choice for continuous mobile sensing.

2.1 Smartphone-based sensing characteristics

Regardless of the sensing paradigm, the internal operation of mobile sensing applications commonly involves a set of stages that describe the closed loop shown in Figure 2.2, consisting in: (a) *Sensor reading* that involves selecting, configuring, and requesting data to sensors, (b) An *optional preprocessing* for discarding uninteresting data like outliers or reducing noise, (c) *Feature extraction* for obtaining features, i.e., more computationally efficient representations of data, (d) *Classification* of extracted features into classes of special interest, for example human activity aspects that might feed machine learning strategies, and (e) *Post-processing* for offering feedback to user, launching network communication, triggering another sensing-classification chain with the outcome of classification acting as feedback information for a more precise operation of sensors, or for performing further processing. At each stage, the level of context-awareness of the mobile platform is increased, requiring specific algorithmic solutions to perform the above described functions. The design of such solutions is not straightforward, as there are a number of factors that impact the operation of each

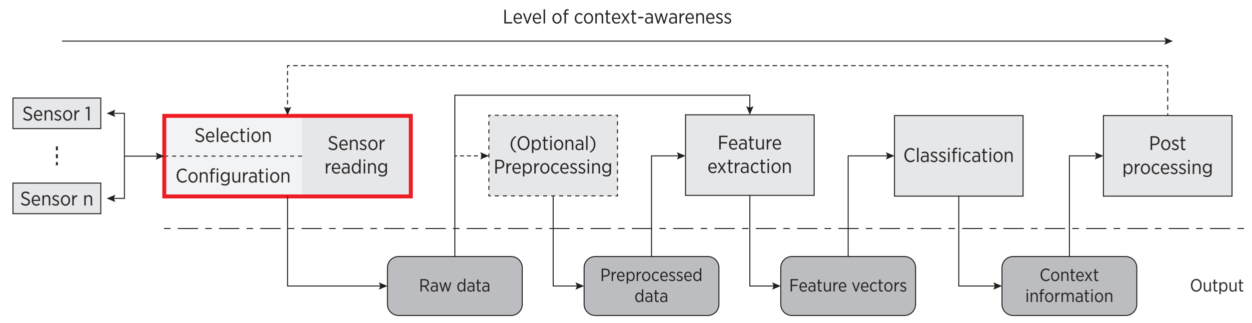

Figure 2.2: The different stages of mobile sensing applications produce output, shown in the lower gray rounded boxes, whose level of information is increasing at each stage. The post-processing stage might launch a new sensor reading chain as shown by the top dashed line.

stage. This is the case of the dynamics of user's context which depends on changes produced in the environment, the existing constraints in mobile platforms, and the privacy concerns of personal data manipulation that, as a whole, compromise the sensor readings, classification and post-processing of data. Similarly, the implementation of these applications is hindered by priorities and additional requirements like the latency at which a context change is detected, the accuracy of the inferred context, and the implicit rational usage of energy resources.

Despite the fact there is a common understanding of how to deploy the aforementioned stages in mobile sensing applications, the support for continuous sensing is still an open research problem due to the power demands of long-term sensing and the inherent resource-constrained characteristics of mobile devices. Although modern smartphones feature increasing computing power, memory and longer battery life, the requirements of complex mobile sensing applications represent a great challenge for mobile platforms, especially to the battery consumption, whose typical capacity is on the order of thousands of mAh, growing only 5-10% each year [Ma et al., 2012, Evarts, 2015]. This growing rate is not at the same pace than the advances in hardware and software components of mobile platforms, which in every new generation of mobile devices impose an ever growing energy demand.

At the same time, it is important to note that since their initial conception, mobile platforms described a layered architecture with a clear focus on managing the heavy interaction with user. because of this, the support for background running processes and for exploiting the power management capabilities existing at hardware level of sensors was discarded, leaving out the power-efficiency of the sensing dimension [Priyantha et al., 2011]. As a consequence, efficient implementations of power-aware mechanisms that do not jeopardize the accuracy requirements of mobile sensing applications still remain an open challenge, which is augmented by the computational and power constraints of mobile devices, as well as the long-term and near real-time requirements of these applications.

2.2 Hypothesis

As previously described, long-time operation of mobile sensing applications remains as an open challenge, due to the inherent trade-off between the energy consumed when accessing to sensors and the accuracy of the activity being tracked. In simpler terms, the more accuracy obtained when accessing sensors with high frequency, the higher energy consumption. Nonetheless, this research work considers that the dynamics of context information inferred from sensors data is of remarkable value for adapting sensory operations and producing power savings. Thus, the hypothesis pursued in this work states that *intelligent policies produced through context information built from sensors data can be employed to reduce the energy consumption in a mobile device when performing continuous sensor readings.*

In a deeper description, a smart policy is a special rule that defines how sensors should be accessed in order to reduce the energy consumption and achieve mobile app requirements. This research work aims to employ data coming from GPS and inertial sensors in order to obtain context information in terms of mobility patterns, useful to reduce the energy consumption generated by LBS's [2].

2.3 Problem statement

In order to prove the aforementioned hypothesis, this research work identifies the existence of two interrelated problems, which are described as follows. The first one is depicted as a *pattern identification problem* and is focused on detecting changes in the mobility pattern of user from sensors data. Such pattern represents the context information that is helpful to add *smartness* to the policies that adapt sensory operations.

Formally: Given a set $V = \{v_1, v_2, \dots, v_n\}$ of data values read from sensor S in the time interval $T = [t_1, t_2]$, find the behavior pattern $Pattern_S$ that represents the activity of user.

$$PatternIdentifier(V) \longrightarrow Pattern_S \in Patterns \tag{2.1}$$

Where $Patterns$ is a set of patterns that represent an interesting state in the user mobility data, namely $\{no\_movement, walking, running, vehicle\_transportation\}$.

[2]LBS, Location Based Service.

On the other hand, the *policy generation problem* is related to the selection and configuration of proper sensors for keeping the user location tracking, while at the same time reducing energy consumption. For this process, it is also important to consider the level of the smartphone's battery and the accuracy requested by the mobile application.

Formally: Given the detected pattern $Pattern_S$ in data from sensor S, parameters for assigning weight to energy eh, and physical constraints status pc of a mobile device, find a policy to adapt the duty cycle of sensors.

$$PolicyGeneration(Pattern_S, eh, pc) \longrightarrow DutyCycle_S \tag{2.2}$$

2.4 Objectives

Main objective

To reduce energy consumption in the mobile sensing apps, which perform continuous sensor readings, through self-adapting power-aware policies generated from context information obtained from sensors data.

Particular objectives

- To identify mobility patterns from context information obtained from an inertial sensor (accelerometer) and location providers (GPS, WPS).
- To generate policies for a self-adapting sensors' usage from identified mobility patterns, accuracy and energy requirements of mobile application, and status of mobile device's constraints.
- To ease the development of mobile sensing applications that require user location tracking, i.e., LBS, isolating the complexity of sensors' access and the associated efficient energy management.

2.5 Contributions

- A mechanism for detecting mobility patterns from the data read by sensors of mobile devices (especifically GPS and accelerometer).
- A mechanism for generating policies for accessing sensors. Such mechanism employs application requirements (energy and precision hints), level of mobile device constraints, and user's context information (using the pattern detected by previous mechanism). The produced policies will allow to perform a smarter usage of smartphone's sensing infrastructure in continuous sensor readings, reducing the energy consumption.

- A middleware that implements the previously described mechanisms, easing the development of mobile sensing applications.

3 State of art analysis

Research on power consumption of mobile devices started since their conception and introduction to the market, and has produced several power management strategies that show an evolution on their complexity and flexibility. Early techniques consisted on generic mechanisms for accessing platform components, mostly at hardware level, like adapting their voltage and frequency of operation, as well as modifying their active power mode [Mayo et al., 2003, Lorch and Smith, 1998, Benini et al., 2000]. Current strategies are focused in specific design issues of particular sensors, tasks, and scenarios where both hardware and software scale their features and energy usage according to changes in the mobile platform workload [Lane et al., 2010, Hoseini-Tabatabaei et al., 2013]. Intuitively, if there are no hard power consumption constraints, the software stack can use hardware components at maximum performance. On the other hand, the mobile platform must react accordingly and modify the hardware usage by employing a set of policies that helps to reduce energy consumption and still achieve the accuracy and performance requirements of mobile applications. Leveraging on context information plays a key role to optimize the *sensor reading* stage (represented in Figure 2.2), reducing the number of sensor invocations, and producing such smart and dynamic adaptation in the sensing dimension. On the other hand, ignoring any source of context information at the sensor reading stage will avoid dynamic adaptations, leading to a fixed hardware usage and neglecting power savings. As a result, there is an inherent power consumption-accuracy trade-off when accessing and consuming sensor data [Sim et al., 2014, Rachuri et al., 2012].

Power-aware sensing refers to a set of techniques and methodologies aimed at continuously monitoring sensor data over long periods of time under the computing, storage, and power constraints of typical mobile devices. Roughly speaking, the power-aware sensing is mostly enclosed into the sensor reading stage of mobile sensing applications, abstracting the complexity of sensors management, and delivering collected data while considering the power constraints of these devices. It is important to remark that this goal is frequently achieved by executing the rest of the stages internally, in order to identify context information that is useful for feedback purposes and a smarter usage of sensors. A revision of relevant works in literature reveals three complementary approaches at different level of abstraction of the smartphone device: pure hardware, hardware-software, and pure software approaches, based on the classification of generic power-aware computing techniques described in [Ranganathan, 2010]. The proposed

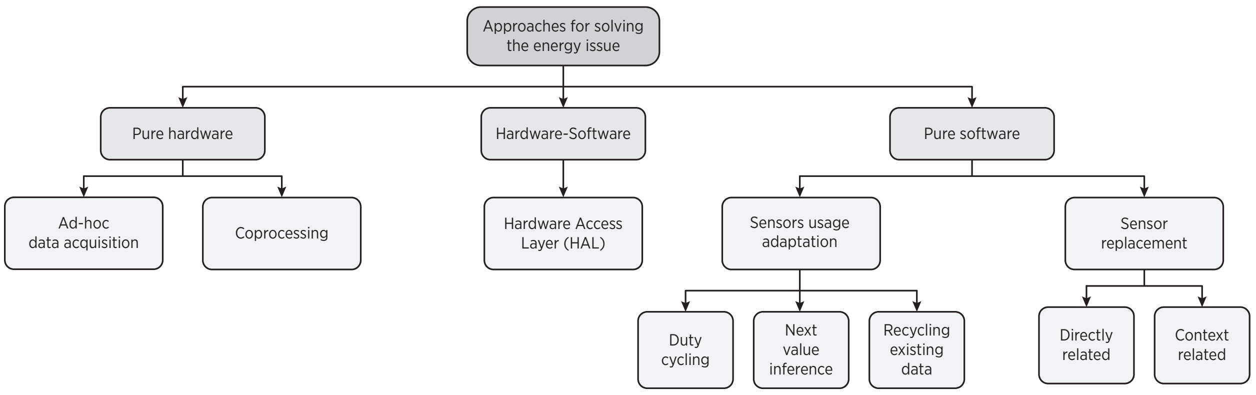

Figure 3.1: The taxonomy of solutions for power-aware smartphone sensing is divided into three categories: pure hardware, hardware-software and pure software.

taxonomy, and definition of each of the categories and their specializations, will be described in next subsections. Figure 3.1 presents a taxonomy of solutions for power-aware smartphone sensing, with the pure hardware, hardware-software and pure software approaches as the main classes, briefly described in the next paragraphs. These approaches incorporate policies with embedded algorithms that analyze simple or complex context information for controlling the power consumption, producing *recipes for power-aware sensing* in mobile devices. It has been identified that the support for these policies at system level of a mobile platform facilitates the analysis, control, and cross-layer coordination and information sharing for reducing the power consumption, which is the reason why power-aware sensing mechanisms should be supported throughout all layers of the mobile platform [Ranganathan, 2010]. Figure 3.2 illustrates the distribution of the different approaches across the layers of a mobile platform, highlighting the increasing support in the higher layers for reacting accordingly to the dynamics of context information through more complex and robust machine learning mechanisms. The distinctive characteristics of the different approaches are as follows:

- The **pure hardware approach** is the lowest level at which power-aware optimizations could be performed. It involves the selection of power-aware sensors and embedded components to deliver physical data to upper layers of mobile platform, as well as the definition of the different power modes of the hardware components through techniques like Dynamic Voltage and Frequency Scaling (DVFS) [Priyantha et al., 2011, Choi and Cha, 2010]. Such power modes define a static behavior of hardware components, which can be adapted through hardware control points exported to upper layers, as in [Priyantha et al., 2011, Choudhury et al., 2008, Apple, 2013].

- The **hardware-software approach** abstracts fine grain details and isolates the

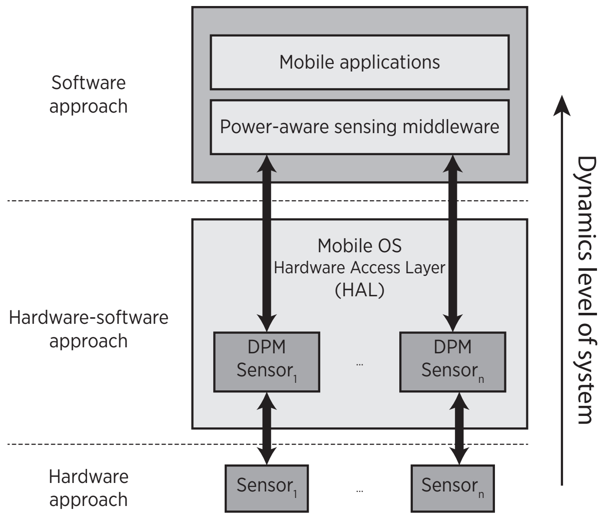

Figure 3.2: The approaches distributed across the mobile platform stack are more adaptive to system's dynamics on the higher layers.

usage of hardware components for mobile applications and upper platform layers. This isolation is performed by applying system-wide policies that coordinate the operation and interaction of the whole set of sensors according to dynamic changes in their workload and global status of the mobile platform (like battery level). The hardware-software approach is aware of the hardware components of the mobile platform, and is able to define DPM mechanisms for turning sensors on and off, and adapt their configuration parameters, as proposed in [Ra et al., 2012, Zhuang et al., 2010].

- The **pure software approach** is aimed at modeling, identifying and even predicting details about context information and its dynamic changes from sensor data, in order to define smart mechanisms for power-aware adaptation of the hardware components usage, like the implementations by [Chon et al., 2014, Yurur et al., 2014, Ma et al., 2009]. Because of the fully context-awareness that this approach achieves, it is able to obtain high levels of flexibility for adaptive management of sensors, and put the context information detected from sensors data at service of the whole mobile platform.

It is important to highlight that although the presented taxonomy allows to classify the power-aware smartphone-based sensing techniques, the relevance of this problem makes that in practice many of the solutions combine different strategies pursuing an integral and better power management. This is a suggestion of the complexity of the problem itself and an evidence of the evolution and adoption of these complementary

Year	Name	Mobile platform	Description	Approach
[Choudhury et al., 2008] 2008	*The Mobile Sensing Platform*	Intel iMote with custom hardware	User motion	Pure hardware
[Priyantha et al., 2011] 2011	*LittleRock*	Not specified	User motion	Pure hardware
[Apple, 2013] 2013	*Apple iPhone 5s*	Apple iPhone 5s and newer	User motion	Pure hardware
[Zhuang et al., 2010] 2010	Not given	Android G1 (Android 1.5)	User location	Hardware-Software
[Ra et al., 2012] 2012	Not given	Samsung SGH-i917 (Windows Phone 7.5)	Generic sensors access	Hardware-Software
[Constandache et al., 2009] 2009	*EnLoc*	Nokia N95 (Symbian 60)	User location	Pure software
[Abdesslem et al., 2009] 2009	*SenseLess*	Nokia N95 (Symbian 60)	User location	Pure software
[Wang et al., 2009] 2009	*EEMSS*	Nokia N95 (Symbian 60)	User location, user activity	Pure software
[Kjaergaard et al., 2009] 2009	*EnTracked*	Nokia N95 (Symbian 60)	User location	Pure software
[Ma et al., 2009] 2009	*iLoc*	Simulation	User location	Pure software
[Paek et al., 2010] 2010	*RAPS*	Nokia N95 (Symbian 60)	User location	Pure software
[Kim et al., 2010] 2010	*SensLoc*	Simulation	User location	Pure software
[Perez et al., 2010] 2010	*G-Sense*	Simulation	User location, user activity	Pure software
[Lin et al., 2010] 2010	*A-Loc*	Simulation and Android G1	User location	Pure software
[Lu et al., 2010] 2010	*Jigsaw*	Nokia N95, Apple iPhone	User location, user activity	Pure software
[Chon et al., 2011] 2011	*SmartDC*	HTC Desire (Android 2.1)	User location	Pure software
[Paek et al., 2011] 2011	*CAPS*	Nexus One (Android 1.5-2.2)	User location	Pure software
[Srinivasan				

techniques. Furthermore, ideas proved to be working efficiently in software soon or later are implemented directly in hardware to boost efficiency even further.

Despite the different approaches for achieving power-aware sensing may be applicable to any set of sensors, in practice they have been mainly employed for optimizing power consumption of location providers, like the GPS and Wi-Fi Positioning System (WPS), in continuous Location-based Services (LBS). The reason behind this trend is twofold. On one hand, the optimization of energy hungry hardware elements like the GPS and other radio interfaces (i.e., GPRS, Bluetooth, etc.), allows the different power-aware sensing techniques to prove their effectiveness under the most strident conditions in terms of energy consumption than those observed in applications that make use of lower power consumption sensors. On the other hand, location information in smartphones is required by a wide variety of applications and therefore, the optimization of energy resources of the GPS sensor reading process (which is the core enabler of LBS) is required. Nonetheless, it is noteworthy that the principles of some of the different proposed solutions can be tailored and implemented in other application niches employing different sensors.

Table 3.1 summarizes a set of works found in literature aimed at solving the power consumption issue in mobile sensing applications in the context of the presented taxonomy. As it can be noted, the most common strategy followed in the literature is the pure software approach, due to its flexibility and fine-tuning features, which will be described in next sections of this paper. Because of the heterogeneity of target features, main purpose, and mobile platforms where the mechanisms proposed in each work were implemented, it is not straightforward to perform a fair comparison among them [Vallina-Rodriguez and Crowcroft, 2013, Neely et al., 2008]. For example, the solutions may target a different set of sensors or the techniques for analyzing context information produce outputs with different semantic meaning. Also there is no a clear consensus for measuring the energy benefits of proposed solutions; in this regard two main strategies have been followed. The first and most common strategy is to perform a comparison in terms of mW between energy consumed by a solution and the mobile platform or a similar approach. However, even inside this strategy the implementations are not homogeneous and the proposed solutions employ different methods for measuring energy consumption in mobile devices, namely highly-invasive, moderately-invasive, and non-invasive techniques, as described in [Abreu and Villapol, 2012]. The second strategy performs the comparison in terms of battery life (e.g., time to drain) because, according to [Kim et al., 2014], several physical characteristics of batteries impact on their discharging cycle. The most notorious aspect is that the actual available energy is reduced when the battery discharges, a behavior that is ignored in the energy consumption models included in most of the reviewed solutions.

The next sections of the document present a description of relevant aspects about the different approaches of the proposed taxonomy. Important concepts about the pure hardware and hardware-software approaches have been already discussed in existing works [Priyantha et al., 2011, Ranganathan, 2010, Lorch and Smith, 1998, Benini et al., 2000]. Hence, this survey includes a brief description about them, as well as the analysis of associated specializations found in state-of-art works. Subsequently, a detailed analysis of the features offered by the pure software approach is presented. This analysis includes its fundamental aspects, its advantages and possibilities when compared against other approaches, as well as the description of its flexibility for modeling and adapting to dynamic changes in context information, which makes it ideal for fitting the system behavior according to user profile.

3.1 Pure hardware approach

The fundamental idea behind the pure hardware approach is the selection of power-aware hardware elements for providing physical data to upper layers of the mobile platform, as well as the definition of mechanisms to adapt the hardware input parameters, like DVFS. These mechanisms allow the creation of control points for manipulating hardware usage, which are known as power modes [Ranganathan, 2010, Lorch and Smith, 1998, Benini et al., 2000]. The mobile platform keeps the list of power modes fixed and instruct sensors to work isolated from others. In this way, the hardware components obey a static behavior defined only through power modes, whose control points are exported to upper layers of the mobile platform.

The set of works under this approach typically involve the design of specific hardware to exploit the aforementioned features. This design implies the isolation of the components associated with the measurement of physical signals inside a new unit with a dedicated low power processor (LPP). Since the mobile OS isolates the access to these hardware components, mobile applications remain agnostic about the new hardware unit and can consume its power-aware sensing services without modifying their inner logic. The basic behavior of this unit requires sensors to report their readings to the embedded LPP, which is capable of executing light preprocessing operations like noise filtering. While the hardware unit performs the reading and filtering stages, the rest of the smartphone hardware platform is able to reach its sleep mode, producing power savings. Table 3.2 shows some of the most representative works under the pure hardware approach. Depending on the ability to process data and discover contextual information autonomously, the pure hardware approach can be implemented in two different ways, namely the *ad-hoc data acquisition* and the *co-processing* variants, as shown in Figure 3.1.

Name	**Variant**	**Sensors involved**
LittleRock [Priyantha et al., 2011]	Ad-hoc data acquisition	Accelerometer, compass, gyroscope
Apple iPhone 5s [Apple, 2013]	Coprocessing	Accelerometer, compass, gyroscope
The Mobile Sensing Platform [Choudhury et al., 2008]	Coprocessing	Accelerometer, barometer, compass, humidity, light, microphone, temperature

Table 3.2: Representative works under the pure hardware approach. More context-aware features are being deployed directly into hardware components of mobile platforms.

The ***ad-hoc data acquisition*** variant is restricted to serve the different mobile application requests for raw sensor data, discarding the analysis and extraction of higher level information. For instance, *LittleRock*, proposed by Priyantha *et al.*, in [Priyantha et al., 2011], defines a specific unit with embedded sensors powered directly from battery, allowing the main circuitry of the smartphone to reach the sleep mode while performing sensor readings. Once a special event defined by a mobile application is detected in data coming from sensors, *LittleRock* is able to wake up the phone for further processing. *LittleRock* is faster than the phone itself to collect a single sample of data from sensors, due to the complexity of the software stack that smartphone's main processor has to handle.

Besides the raw data delivery, the ***co-processing*** variant adds support to hardware layers for light data processing and detection of higher level information, producing context-awareness at hardware level with low power consumption. In this sense, the sensing facilities of the hardware platform are transformed into smart sensors [Gervais-Ducouret, 2011]. The embedded LPP of the sensing unit must be powerful enough to perform such analysis.

Smartphone manufacturers have begun to integrate into the device's hardware full suites of sensing capabilities. For instance, Apple has implemented a platform re-design of the *iPhone* smartphone, which starting at *5s* version features an isolated sensing unit able to process sensor data and deliver fitness information like the amount of steps and distance covered by user, as well as the type of activity performed, namely, stationary, walking, running, automotive, cycling, and unknown [Apple, 2013]. In addition to this hardware platform update, the Software Development Kit (SDK) of iPhone exposes this functionality through a software Application Programming Interface (API) denominated the *CoreMotion Framework* that enables mobile applications to request and consume the fitness information. A similar co-processing mechanism is implemented by Choudhury *et al.*, in the *Mobile Sensing Platform* (*MSP*) [Choudhury et al., 2008] which employs ad-hoc components for building a custom mobile sens-

Reference	Produced HAL	Sensors involved
[Ra et al., 2012]	API for selecting processor for execution of arbitrary tasks	Accelerometer, microphone
[Zhuang et al., 2010]	Automatic substitution, suppression, piggybacking, and adaptation of location providers	GPS, wireless interface

Table 3.3: Representative works under the hardware-software approach. The produced hardware interfaces improve the overall efficiency of the mobile platform.

ing platform with a LPP able to perform light classification tasks for identifying user activity.

A common aspect taken into account in the design of pure hardware approach solutions is the power consumption - usage generality trade-off that emerges when hardware designers are unable to foresee the needs of any future mobile application and implement power-aware mechanisms to support them directly in the circuits. Because of this limitation, the effort of designers is focused on implementing general features that can be employed by many mobile sensing applications, instead of very specific features meaningful only for a few of them.

3.2 Hardware-software approach

The hardware-software approach is aimed at defining system-wide policies for deciding when to turn sensors on and off, or when to switch to a different power mode of a given hardware component. In this level, the hardware-software approach is aware of the existence of different sensors and coordinates the basic operations and interactions between them in a DPM fashion, being able to abstract fine grained parameters into a coarser grained set, isolating the hardware usage for mobile applications and elements in upper platform layers. Unlike the pure hardware approach, the hardware-software approach offers a basic understanding of global activities being performed by the mobile device, and also an implicit observance of dynamic changes in the workload of hardware components for deciding when to adapt hardware operation. Because of the coupled interaction with hardware components of the platform, hardware-software approach solutions are produced as HAL's of the mobile OS, as previously illustrated in Figure 3.2, or low level middlewares that build an interface that abstract the access to sensors and isolates their complex management mechanisms. Table 3.3 presents a few instances of works that follow this approach this approach.

An example of the hardware-software approach is the inclusion of software mechanisms that leverage a dedicated LPP for accessing and manipulating sensors data. The software mechanisms allow to build the basic global status of the mobile platform, which is employed for manipulating the hardware through the control points exposed by sensors. Since the LPP can be powered independently of smartphone's main circuitry, it can execute tasks while the rest of the mobile platform can reach the idle state, producing potential energy savings. The work presented by Ra *et al.*, in [Ra et al., 2012] leverages the presence of such LPP's in the latest smartphones, putting their computing facilities at service of mobile applications through a layered API. Any processor with a small wake-up transition delay could be suitable to be used as a LPP, and will achieve the highest power savings when executing the most frequent and lightest computations, like sensor sampling requests and simple filtering.

Zhuang *et al.*, presented in [Zhuang et al., 2010] a hardware-software approach middleware for LBS's execution based on the next four key design principles: (a) *Sensor Substitution*: It allows on-the-fly substitution of current location provider with another with lower energy consumption and an accuracy level enough for the mobile application requirements, (b) *Sensor Suppression*: Avoids usage of any location provider by leveraging data from energy cheaper sensors, like the accelerometer, for revealing when user is static, (c) *Sensor Piggybacking*: It allows to serve location updates to simultaneous LBS's, and (d) *Sensor Adaptation*: It reacts to low battery levels to adapt system-wide sensing parameters like time and distance. For identifying places where it is possible to substitute a location provider, it relies on a novel concept referred to as the *M-Area* which defines a geographical zone along with information about the available location providers on it. The size of these areas involves a trade-off between the sensor substitution events and the storage space needed to save their information.

3.3 Pure software approach

Thanks to the context information obtained from sensor data, the pure software approach can bring activity awareness to the mobile platform and make informed decisions for fully dynamic power-aware sensors management as illustrated in Figure 3.2, where the pure software approach is at the top of the hierarchy. Typically, pure software approach solutions are powered up by a set of classifiers fed with sensors data that individually identify basic aspects or produce abstractions about user's activity. By combining the outcome obtained by these classifiers, it is possible to build an enriched inference of the high level activity performed by the user, for example mobility patterns from location data, human activity from sounds detection, and human activity from inertial sensors data. Such user activity aspects are the fundamental elements for defining sensor management policies. In this way, the pure software approach is able to make the mobile platform aware about user activity and even adapting accordingly to changes on its profile.

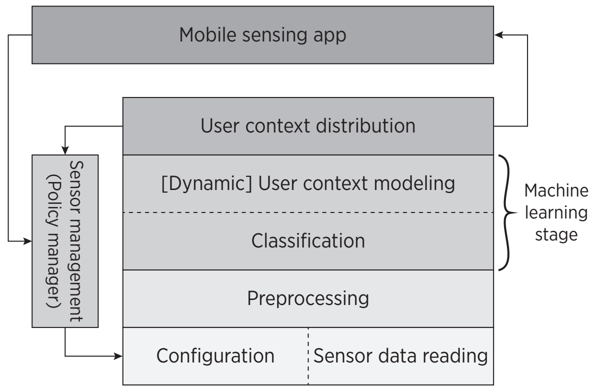

Figure 3.3: A generic structure of a middleware following a pure software approach defines explicit modules for selecting and configuring sensor access, and classification and machine learning stages.

Solutions following this approach can only observe the details of the hardware platform exposed by the hardware access or lower layers of the mobile OS. A practical interpretation can be expressed as follows: these lower layers know how to turn circuits on and off, but are unable to define when; whereas the higher software layers are flexible and can dynamically adapt to changes in user context, knowing when adapt the sensors operation, but delegating how to do it to the lower layers. Typically, pure software approach solutions are implemented through a layered middleware with classification and machine learning modules embedded on it. Figure 3.3 shows a generic structure for pure software approach middlewares, where the different stages of mobile phone sensing described in Section **??** can be observed. This middleware is placed between running mobile applications and the sensing and communication layers of the mobile platform, abstracting its sensing and communication facilities [Yurur et al., 2014]. Figure 3.3 is also helpful to identify that the outcome of classification and machine learning stage can be distributed to mobile sensing applications as well as being employed as feedback information for defining policies for sensors management. In this regard, the feedback information allows to build the closed loop described in the stages of mobile sensing applications, shown in Figure 2.2, which permits the mobile platform to react to changes detected in context by configuring and adapting sensors usage according to such changes, mobile application requirements, and mobile platform constraints.

References

[Abdesslem et al., 2009] Abdesslem, F. B., Phillips, A., and Henderson, T. (2009). Less is more: energy-efficient mobile sensing with senseless. In *Proceedings of the 1st ACM workshop on Networking, systems, and applications for mobile handhelds.*, pages 61–62.

[Abreu and Villapol, 2012] Abreu, D. P. and Villapol, M. E. (2012). Measuring the energy consumption of communication interfaces on smartphones using a moderately-invasive technique. In *2012 Global Information Infrastructure and Networking Symposium, GIIS 2012*, pages 1–6.

[Álvarez De La Concepción et al., 2014] Álvarez De La Concepción, M. a., Soria Morillo, L. M., Gonzalez-Abril, L., and Ortega Ramírez, J. a. (2014). Discrete techniques applied to low-energy mobile human activity recognition. A new approach. *Expert Systems with Applications*, 41:6138–6146.

[Apple, 2013] Apple (2013). Core Motion Framework Reference.

[Benini et al., 2000] Benini, L., Bogliolo, A., and De Micheli, G. (2000). A survey of design techniques for system-level dynamic power management. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 8(3):299–316.

[Campbell and Choudhury, 2012] Campbell, A. and Choudhury, T. (2012). From smart to cognitive phones. *IEEE Pervasive Computing*, 11:7–11.

[Chen and Kotz, 2000] Chen, G. and Kotz, D. (2000). A Survey of Context-Aware Mobile Computing Research. Technical report.

[Choi and Cha, 2010] Choi, J. and Cha, H. (2010). System-level power management for system-on-a-chip -based mobile devices. *IET Computers & Digital Techniques*, 4(5):400.

[Chon et al., 2014] Chon, Y., Kim, Y., Shin, H., and Cha, H. (2014). Adaptive duty cycling for place-centric mobility monitoring using zero-cost information in smartphone. *IEEE Transactions on Mobile Computing*, 13(8):1694–1706.

[Chon et al., 2011] Chon, Y., Talipov, E., Shin, H., and Cha, H. (2011). Mobility prediction-based smartphone energy optimization for everyday location monitoring. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems - SenSys '11*, page 82.

[Choudhury et al., 2008] Choudhury, T., Borriello, G., Consolvo, S., Haehnel, D., Harrison, B., Hemingway, B., Hightower, J., Klasnja, P., Koscher, K., LaMarca, A., Landay, J. a., LeGrand, L., Lester, J., Rahimi, A., Rea, A., and Wyatt, D. (2008). The mobile sensing platform: An embedded activity recognition system. *IEEE Pervasive Computing*, 7:32–41.

[Constandache et al., 2009] Constandache, I., Gaonkar, S., Sayler, M., Choudhury, R. R., and Cox, L. (2009). EnLoc: Energy-efficient localization for mobile phones. In *Proceedings - IEEE INFOCOM*, number 4, pages 2716–2720.

[Donohoo et al., 2014] Donohoo, B. K., Ohlsen, C., Pasricha, S., Xiang, Y., and Anderson, C. (2014). Context-aware energy enhancements for smart mobile devices. *IEEE Transactions on Mobile Computing*, 13(8):1720–1732.

[Ericsson, 2015] Ericsson (2015). Ericsson Mobility Report. Technical Report November.

[Evarts, 2015] Evarts, E. C. (2015). Lithium batteries: To the limits of lithium. *Nature*, 526(7575):S93–S95.

[Gervais-Ducouret, 2011] Gervais-Ducouret, S. (2011). Next smart sensors generation. *SAS 2011 - IEEE Sensors Applications Symposium, Proceedings*, pages 193–196.

[Hoseini-Tabatabaei et al., 2013] Hoseini-Tabatabaei, S. A., Gluhak, A., and Tafazolli, R. (2013). A Survey on Smartphone-Based Systems for Opportunistic User Context Recognition. *ACM Computing Surveys*, 45(3):27:1–27:51.

[Khalifa et al., 2015] Khalifa, S., Hassan, M., and Seneviratne, A. (2015). Pervasive self-powered human activity recognition without the accelerometer. In *2015 IEEE International Conference on Pervasive Computing and Communications (PerCom)*, pages 79–86. IEEE.

[Kim et al., 2010] Kim, D. H., Kim, Y., Estrin, D., and Srivastava, M. B. (2010). SensLoc: Sensing Everyday Places and Paths Using Less Energy. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, SenSys '10, pages 43–56, New York, NY, USA. ACM.

[Kim et al., 2014] Kim, M., Kim, Y. G., Chung, S. W., and Kim, C. H. (2014). Measuring Variance between Smartphone Energy Consumption and Battery Life. *Computer*, 47(7):59–65.

[Kjaergaard, 2012] Kjaergaard, M. (2012). Location-based services on mobile phones: Minimizing power consumption. *IEEE Pervasive Computing*, 11:67–73.

[Kjaergaard et al., 2009] Kjaergaard, M. B., Langdal, J., Godsk, T., and Toftkjær, T. (2009). EnTracked : Energy-Efficient Robust Position Tracking for Mobile Devices. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 221–234.

[Lane et al., 2010] Lane, N. D., Miluzzo, E., Lu, H., Peebles, D., Choudhury, T., and Campbell, A. T. (2010). A survey of mobile phone sensing. *IEEE Communications Magazine*, 48(September):140–150.

[Lin et al., 2010] Lin, K., Kansal, A., Lymberopoulos, D., and Zhao, F. (2010). Energy-accuracy trade-off for continuous mobile device location. In *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*, page 285.

[Lorch and Smith, 1998] Lorch, J. and Smith, A. (1998). Software strategies for portable computer energy management. *IEEE Personal Communications*, 5(3):60–73.

[Lu et al., 2010] Lu, H., Yang, J., Liu, Z., Lane, N. D., Choudhury, T., and Campbell, A. T. (2010). The Jigsaw continuous sensing engine for mobile phone applications. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems - SenSys '10*, page 71.

[Ma et al., 2012] Ma, X., Cui, Y., and Stojmenovic, I. (2012). Energy efficiency on location based applications in mobile cloud computing: A survey. In *Procedia Computer Science*, volume 10, pages 577–584.

[Ma et al., 2009] Ma, Y., Hankins, R., and Racz, D. (2009). iLoc: a framework for incremental location-state acquisition and prediction based on mobile sensors. In *Proceeding of the 18th ACM conference on Information and knowledge management - CIKM '09*, page 1367, New York, New York, USA. ACM Press.

[Man and Ngai, 2014] Man, Y. and Ngai, E. C. H. (2014). Energy-efficient automatic location-triggered applications on smartphones. *Computer Communications*, 50:29–40.

[Mayo et al., 2003] Mayo, R., Mayo, R., Ranganathan, P., and Ranganathan, P. (2003). Energy consumption in mobile devices: Why future systems need requirements-aware energy scale-down. In *Power - Aware Computer Systems*, pages 26–40.

[Mazilu et al., 2013] Mazilu, S., Blanke, U., Calatroni, A., and Tröster, G. (2013). Low-power ambient sensing in smartphones for continuous semantic localization. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 8309 LNCS, pages 166–181, Ireland. Springer.

[Morillo et al., 2015] Morillo, L., Gonzalez-Abril, L., Ramirez, J., and de la Concepcion, M. (2015). Low Energy Physical Activity Recognition System on Smartphones. *Sensors*, 15(3):5163–5196.

[Neely et al., 2008] Neely, S., Stevenson, G., Kray, C., Mulder, I., Connelly, K., and Siek, K. a. (2008). Evaluating pervasive and ubiquitous systems. *IEEE Pervasive Computing*, 7(3):85–88.

[Paek et al., 2010] Paek, J., Kim, J., and Govindan, R. (2010). Energy-efficient rate-adaptive GPS-based positioning for smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*, volume 223-224, page 299, New York, New York, USA. ACM Press.

[Paek et al., 2011] Paek, J., Kim, K.-H., Singh, J. P., and Govindan, R. (2011). Energy-efficient positioning for smartphones using Cell-ID sequence matching. In *Proceedings of the 9th international conference on Mobile systems, applications, and services - MobiSys '11*, page 293, New York, New York, USA. ACM Press.

[Perez et al., 2010] Perez, A. J., Labrador, M. a., and Barbeau, S. J. (2010). G-Sense: A scalable architecture for global sensing and monitoring. *IEEE Network*, 24(August):57–64.

[Perez-Torres and Torres-Huitzil, 2012] Perez-Torres, R. and Torres-Huitzil, C. (2012). A power-aware middleware for location & context aware mobile apps with cloud computing interaction. In *Proceedings of the 2012 World Congress on Information and Communication Technologies, WICT 2012*, pages 691–696.

[Priyantha et al., 2011] Priyantha, B., Lymberopoulos, D., and Liu, J. (2011). Little-Rock: Enabling energy-efficient continuous sensing on mobile phones. *IEEE Pervasive Computing*, 10:12–15.

[Ra et al., 2012] Ra, M., Priyantha, B., Kansal, A., and Liu, J. (2012). Improving Energy Efficiency of Personal Sensing Applications with Heterogeneous Multi-Processors. In *The 14th International Conference on Ubiquitous Computing*, pages 1–10.

[Rachuri et al., 2012] Rachuri, K. K., Mascolo, C., and Musolesi, M. (2012). *Mobile Context Awareness*. Springer London, London.

[Ranganathan, 2010] Ranganathan, P. (2010). Recipe for efficiency. *Communications of the ACM*, 53(4):60.

[Sim et al., 2014] Sim, J., Lee, Y., and Kwon, O. (2014). Context-aware enhancement of personalization services: A method of power optimization. *Expert Systems with Applications*, 41(13):5702–5709.

[Srinivasan and Phan, 2012] Srinivasan, V. and Phan, T. (2012). An accurate two-tier classifier for efficient duty-cycling of smartphone activity recognition systems. In *Proceedings of the Third International Workshop on Sensing Applications on Mobile Phones - PhoneSense '12*, pages 1–5.

[Vallina-Rodriguez and Crowcroft, 2013] Vallina-Rodriguez, N. and Crowcroft, J. (2013). Energy management techniques in modern mobile handsets. *IEEE Communications Surveys and Tutorials*, 15(1):179–198.

[Wang et al., 2009] Wang, Y., Lin, J., Annavaram, M., Jacobson, Q. a., Hong, J., Krishnamachari, B., and Sadeh, N. (2009). A framework of energy efficient mobile sensing for automatic user state recognition. In *Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 179–192.

[Yurur et al., 2014] Yurur, O., Labrador, M., and Moreno, W. (2014). Adaptive and energy efficient context representation framework in mobile sensing. *IEEE Transactions on Mobile Computing*, 13(8):1681–1693.

[Zhang et al., 2013] Zhang, L., Liu, J., Jiang, H., and Guan, Y. (2013). SensTrack: Energy-efficient location tracking with smartphone sensors. *IEEE Sensors Journal*, 13(10):3775–3784.

[Zhuang et al., 2010] Zhuang, Z., Kim, K.-H. K., and Singh, J. J. P. (2010). Improving energy efficiency of location sensing on smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services - MobiSys '10*, page 315.