Sensing Middleware for Easy Development of Energy-efficient Applications

● Toru Kamiwada ● Manabu Nakao ● Eiji Hasegawa

Information services can be more useful if mobile devices are adapted to gather information in our surroundings using built-in sensing functions. Mobile devices, however, run on batteries, and various approaches are taken to tackle the major challenge the sensing and telecommunications mechanism poses in terms of their power consumption. Generally, mobile products tend to benefit from power-saving effects more if their purposes or devices to use are more specific. However, saving energy by specifying the usage is likely to lower the products' versatility when it comes to reapplying them to other uses or devices, making it necessary to develop applications and firmware every time the usage or applied devices change. Against this background, Fujitsu Laboratories has developed sensing middleware technology that makes it possible to choose and continually execute the least power-consuming sensing method simply by using the application to specify what kind of information is required, while the middleware collects and manages the power composition of available sensors. This paper presents this middleware that significantly reduces development cost while increasing the application's portability for mobile sensing.

1. Introduction

Smartphones have widely spread and various services based on cloud computing have been emerging in recent years. In tandem with this, a wide range of information services are rapidly becoming more and more prevalent. In particular, such services are increasingly provided to mobile users; various types of information about the users are captured by a multitude of sensors equipped in their smart devices such as smartphones and tablets, and they receive useful information appropriate to their current situations. For example, the users' location data, generated and gathered by means of the GPS and wireless LAN (WLAN), serve to provide information on facilities or stores in their surroundings, or information on navigation routes. Similarly, sensors capture the users' motions, and make it easier to provide information on ways to stay healthy or prevent surrounding risks. New services like these are being launched one after another.

Moreover, the recent appearance of wearable devices—coming in the forms of a wristband, wristwatch, a pair of glasses, etc.—adds to the aspiration to enrich information services for a better user experience in the mobile environment. These developments in the sensing and notifying features of wearable devices, as well as easy-to-use user interfaces, increasingly allow for the provision of detailed information in various situations, effortlessly.

The information services that deploy the sensing technologies and wearable devices thus are expected to be leveraged more in business situations. Take the industries such as logistics, medical services, construction and retailing, for example—workers may not always be able to handle information devices while they are engaged in some tasks. The sensing technologies and wearable devices will make it possible for workers to enter or browse the necessary data hands-free, without disturbing their work.

However, such devices run on batteries, which may run out quickly if the sensors keep operating all the time. To address this problem, there are many power-saving technologies, but they tend to become hardware- or application-specific to obtain more power-saving effects. This means that the power-saving mechanism has to be developed from scratch every time a new hardware device or application is developed.

Given this as a background, Fujitsu Laboratories has developed a software architecture with a focus on sensing middleware, that is designed to help with the development of applications for continuous sensing without depending on hardware configurations. This paper explains this sensing-middleware-based software architecture.

2. Challenges regarding the sensing operation with mobile devices

There are high expectations for a future expansion in the use of information services as technology advances to allow for continuous sensing using mobile devices. As the smart/wearable devices employed will be powered by batteries, and thus have a limited operational time, insufficient battery performance may result in users' having to disturb their work or adding more tasks to complete in the course of work, as they would need to recharge the batteries so often. Therefore, it is imperative to improve the energy efficiency of sensing technology.

2.1 Conventional technology: Off-loading to a dedicated processor

In one of the existing power-saving technologies used with smartphones, a dedicated low-power-consuming processor, apart from the general-purpose main processor (CPU), is provided to process the data from the sensors, thereby reducing the power consumption (**Figure 1**).

The Human Centric Engine[1] used in Fujitsu smartphones is one example. This engine makes it possible to provide operation support services according to the information obtained via the motion sensor, which continually captures the user's motions to recognize if a mobile device is in use or to get information about the user's walking stability, without consuming much of the battery power. Apple's iPhone[2] and Motorola's smartphone[3] soon followed to adopt a similar configuration using dedicated processors to reduce the power consumption of the sensing operations. This configuration of using a dedicated processor that is connected with many varied sensors is on the increase, and today it is recognized as a "sensor hub."[4),5]

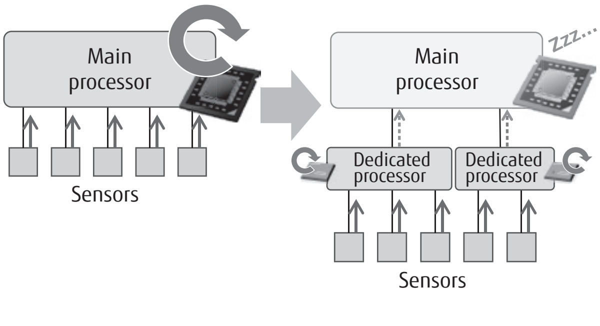

Figure 1
Energy-saving configurations using dedicated processors.

2.2 Challenge: Developing firmware bundling several sensors

As stated above, various sensing services can be made less power-consuming by leveraging dedicated processors. It is possible to further lower the power consumption by minutely controlling the selections of sensors to operate. It should work in such a way that, while keeping on the sensors that need constant monitoring, other sensors may be turned off as default, and turned on only when they are needed. For example, it is significantly power-consuming to constantly search for GPS signals and WLAN access points (APs) to obtain geographical information. This can be reduced if a low-power-consuming acceleration sensor is employed to detect the motion, and while there is no positive output (i.e., the smartphone's position does not change), the sensing for a GPS signal and WLAN APs is turned off. This can be achieved by equipping the device with firmware to control those dedicated processors in place of the main processor. In this way, continuous sensing would be possible in many situations.

However, logical descriptions for the determination of which sensors to select would depend on the applications. For instance, the method of turning off the WLAN while the smartphone is geographically stationary, may be valid for applications based on geofencing, which in turn sends notifications when the person who carries the smartphone moves closer to certain locations. Whereas, if this system is applied to a wireless logger that gathers WLAN AP information, it may result in missing some AP data because certain APs, such as a mobile router, may change their positions while the smartphone remains in one location and WLAN is turned off. As these ways of data processing differ with different applications, it is difficult to equip the smartphone with fixed firmware for dedicated processors. Updating firmware is a labor-intensive task, and it incurs safety risks. Thus, it is not recommended to update the firmware frequently. As a result, the main processor would be employed to execute these application-specific processes. In such a case, the main processor must continually monitor the sensor data to determine whether to turn these other sensors on or off. Even if the dedicated processors were employed to minimize the continued monitoring and save the battery power, the main processor would be running constantly to monitor the data, on which other sensors are turned on and off.

If the combination of sensors may be determined and such sensors may be controlled by a dedicated processor, the main processor will then operate only when the conditions for certain combinations are met. However, as it has been mentioned, the determining logic statements depend on the applications, and this makes it difficult to mount the functionality as fixed firmware. Therefore, there have not been any successful attempts at reducing the main processor's power consumption due to the need to perform this function.

3. Developed technology

To address the challenges stated in the preceding section, Fujitsu Laboratories has developed sensing middleware, which makes it possible to continually look for designated information provided by applications, and execute the sensor control according to individual pieces of sensing data, via the low-power consuming processor. This sensing middleware runs on smartphones in the background, coordinating the applications that send requests for sensing operations with appropriate sensors. The wearable devices with sensors act as sensing nodes in accordance with the command from the middleware, as part of the firmware components. This enables the middleware to process various requests made by the applications on the same firmware. The following are examples of the processes that can be executed by these sensing nodes:

- Operational control of the sensors (on/off)
- Reception of data from the sensors
- Computation based on the sensor data (matching, etc.)
- Recording and retention of the computation results
- Matching according to several computation results

Multiple sensing nodes capable of executing these processes are integrated through the sensing middleware, and the sensing processes are carried out in an energy efficient way over a long period in response to the requests made by the applications by virtue of the distributed processing architecture. The configuration is illustrated in **Figure 2**.

In this architecture, the information relevant to sensing processes is stored in advance, with respect to the variety of sensing features and combinations and determination features of the wearable devices connected to the smartphone, as well as the power that is required to execute the respective sensing and computing. Once the middleware is ready in this respect, the applications may request necessary information from the relevant sensors. An example of a request may be to "notify when a WLAN AP with the service set identifier (SSID) value of XYZ is detected while being on the move." At this point, the application does not need to specify which sensors should be employed or how the data should be processed. It only conveys what kind of information is required.

The middleware, upon receiving this request, interprets the commands to be operating respective sensors to detect "on move status" and "WLAN AP SSID," evaluating the returned outputs to match "on move" and "XYZ" values, and then verifies these matching results. The results of the interpretations will be managed in the list of conditional expressions.[6] Using this list and the previously accumulated information on the wearable devices, the middleware decides the least power-consuming combination of conditional processing and sensors, and it assigns the processes to each device. According to this decision, the conditional expressions are sent to the corresponding devices, thereby commanding the execution of the assigned tasks.

Once this command assignment is complete, the middleware puts the smartphone into sleep mode, and minimizes the power consumption. In the meantime, the wearable devices continue to sense according to the commands. The data from the wearable devices are evaluated against the conditional expressions, and if the outcomes meet the conditions of the sensing request, the middleware wakes up the smartphone and announces that the request has been satisfied. Thus,

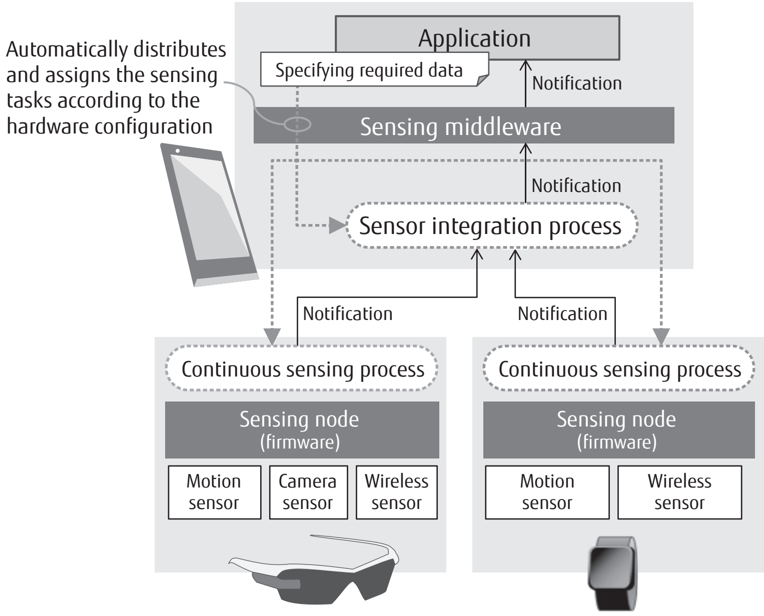

Figure 2
Configuration of developed technology.

while the evaluation process will be executed each time the sensor detects the on-move status or identifies any WLAN APs, this architecture does not allow the smartphone to be woken up until the conditions requested by the application—a WLAN AP with the SSID value of XYZ is detected while being on move—are met, and only then is the main processor activated. This allows for significant time-saving in terms of the smartphone's active operation in relation to this sensing request.

Furthermore, this power-saving architecture spares the application from the need to specify the sensor to be used. This means that developers do not have to worry about the hardware on the mobile device during their development processes, as long as the device is equipped with this middleware, as the same applications operate on the basis of the same logic. The applications can be unified irrespective of the hardware configurations, which helps to avoid increasing the person-hours needed for developing versions of the applications to adapt to different devices. Meanwhile, seen from the perspective of the smart and wearable devices, the architecture realizes power saving through the deployment of dedicated processors in executing continued sensing, while the firmware of the wearable devices does not need to be frequently updated even though the sensing requests by applications may vary. This means that the architecture provides developers with the opportunity to save person-hours in their development activities, and to realize flexible matching between applications and hardware configurations, while saving power in the sensing operations according to these combinations.

4. Operation example

To illustrate the use of the middleware, we will describe a case in which a certain hardware configuration is employed in tasks that require different sensing processes (**Figure 3**). The given tasks are (a) checking facility equipment and (b) picking up goods in a warehouse. Both tasks involve a smartphone equipped with the sensing middleware and a wearable device. The wearable device has firmware installed on it that functions as a sensing node. This device is used in both tasks. Each task is provided with an application, to be operated on the smartphone.

For the equipment check, the device scans for the equipment to be checked around the users whenever they stop, and displays the check items if such equipment is identified as being in the users' surroundings. The pieces of equipment have a wireless beacon attached to them, and the wearable device catches the signal using a wireless sensor. The motion

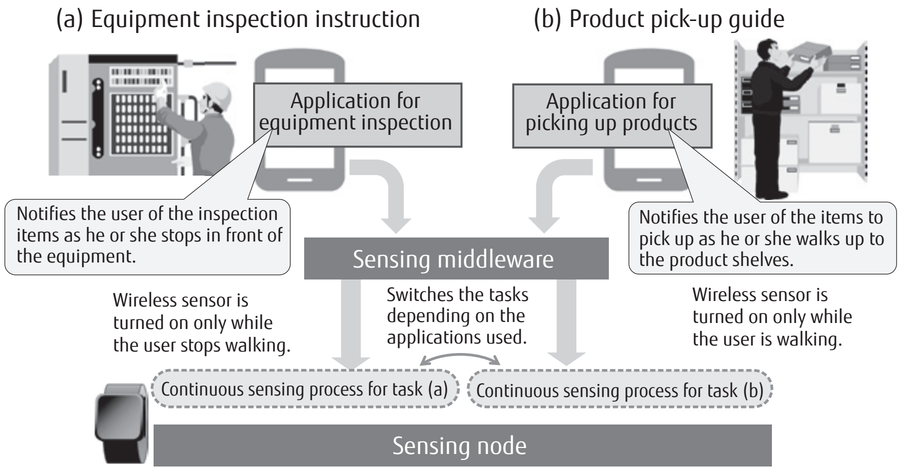

Figure 3
Application example of sensing middleware.

of the users is detected by means of the acceleration sensor embedded in the wearable device. The application used for equipment inspection sends a request to “notify the users if they stop in front of the specified piece of equipment.” Then, the middleware sends a command based on the request to the wearable device, which functions as a sensing node, thereby operating the wireless sensor only when the users stop walking.

For the goods-pickup task, the users receive notifications as they move about in the warehouse whenever they approach a shelf with an attached wireless beacon, and the device prompts the users to pick up the designated goods. The application employed in this task sends a request to “notify the users when they walk up to the beacon with a specific ID,” which the middleware relays to the wearable device. The wireless sensor is then activated only while the users are walking.

As described above, this middleware coordinates with the wearable device under different sensing requests, thereby executing the sensing operations more energy efficiently. Thus, there is no need to modify the wearable devices’ firmware depending on the different applications to connect.

Conventionally, higher energy efficiency has been achieved by developing sensing processes within applications, which also entailed the development of drivers and firmware. In the configuration that uses our middleware, by contrast, the application only needs to specify the type of information required from sensing, allowing for a significant reduction of the person-hours in application development. In fact, by applying this technology, the above-mentioned applications employed in the checking and pick-up tasks needed only one-tenth of the person-hours for the development (**Figure 4**).

5. Conclusion

There are high expectations that information services will be made more useful by leveraging mobile devices for sensing operations. While reducing the power consumption of sensing operations was one of the major challenges, there have been no successful solutions to commonly apply the power-saving technology across different sensing needs and devices. This necessitated the development of individually tailored power-saving mechanisms for each application or service, adding to the person-hours in the development processes.

To address this challenge, Fujitsu Laboratories has developed technology for sensing middleware. The technology has realized middleware that is capable of gathering and managing data from sensor-equipped wearable devices, and by employing these sensors appropriately according to the requests made by various applications. This saves developers the labors of developing new firmware for the wearable devices to adapt to new applications. It also enables applications to be used in a range of hardware configurations. It widens the potential use of provided wearable devices with a

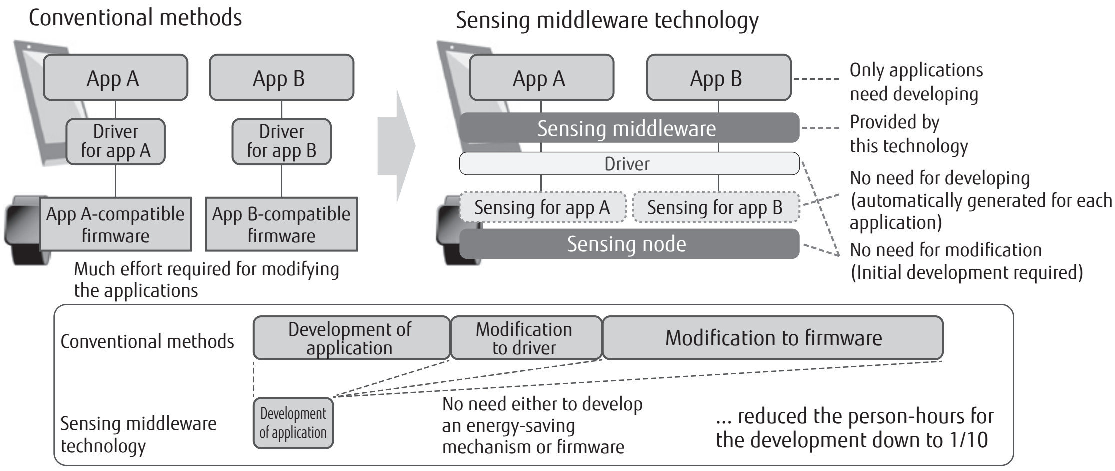

Figure 4
Reducing the person-hours for development.

variety of applications, and at the same time, one application can be employed with many different devices. We have also verified that, in a certain case of sensing tasks, the application development could reduce the person-hours down to one-tenth. Such sensing middleware is expected to enhance productivity in developing applications that employ mobile sensing features, and broaden the potential of information services in the future.

References

1) K. Chujo et al.: Human Centric Engine and Its Evolution toward Web Services. FUJITSU Sci. Tech. J. Vol.49, No.2, pp.153–159 (2013).
http://www.fujitsu.com/global/documents/about/resources/publications/fstj/archives/vol49-2/paper02.pdf
2) Apple: Apple - iPhone 6 - Technology.
https://www.apple.com/iphone-6/specs/
3) Motorola Mobility: X8 mobile computing system for Motorola.
https://www.motorola.com/us/X8-Mobile-Computing-System/x8-mobile-computing-system.html
4) Atmel: Sensor Hub.
http://www.atmel.com/products/drivers-and-sensors/sensor-hub/
5) Qualcomm Technologies: Sensor processing done right: Smart integration of a specialized sensor engine.
https://www.qualcomm.com/news/onq/2014/06/05/sensor-processing-done-right-smart-integration-specialized-sensor-engine
6) T. Sato et al.: Sensing middleware for performing the complex sensing process by minimum power in accordance with adjusted to terminal hardware component. Multimedia, Distributed, Cooperative and Mobile Symposium (DICOMO 2014), pp. 1575–1580 (2012) (in Japanese).

Toru Kamiwada
Fujitsu Laboratories Ltd.
Mr. Kamiwada currently engages in R&D of power-saving on mobile devices, sensing and device connectivity technologies.

Eiji Hasegawa
Fujitsu Laboratories Ltd.
Mr. Hasegawa currently engages in R&D of sensing technology for mobile devices.

Manabu Nakao
Fujitsu Laboratories Ltd.
Mr. Nakao currently engages in R&D of sensing technology.