Abstract
Nowadays, mobile devices are becoming more flexible and rich in functionalities. As already presented in [6] those devices are highly influenced by constraints, mainly regarding power management. In fact, mobile batteries are limited in time and there are no efficient methods able to manage power consumption. Even knowing the device Time To Live (TTL) is currently left to the user experience. In this paper, we presented MPower, a system able to predict the mobile device TTL, providing also the user with suggestions on the optimal device configuration w.r.t. the desired TTL. This allows the user to manage the available power resources, according to his/her needs, avoiding power wasting.

Author Keywords
Mobile, Power modeling

General Terms
Mobile, Context-awareness

ACM Classification Keywords
C.3.f [Special-Purpose and Application-Based Systems]: Signal processing systems.
**Introduction**

Mobile devices, e.g., smartphones, PDAs and tablets, increase in number of functionalities in the recent years, making them appealing to a wide range of users. Thus, even if they are becoming more and more powerful in term of supported computation, they are constrained w.r.t. the power consumption. In fact, batteries are limited in time and their *Time To Live* (TTL) is highly influenced by the current phone state, the user usage profile and external conditions, e.g., signal strength. As a consequence, since these devices are subject to sudden and unpredictable changes of their conditions, a method for computing the device TTL is necessary. A practical way of predicting the TTL is to model the battery discharging curve, since TTL coincides with the time the battery level reaches zero level.

Android OS [1], one of the most popular mobile OS, only provides a percentage as remaining power indicator, leaving to the user experience and knowledge to determine the TTL of its mobile device. In order to know a TTL, the discharge curve is needed, and consequently, a power model of the device. Literature on mobile power consumption estimation provides a wide range of works about power modeling [6]. Most of them use benchmark applications on a single device to gather data, providing model which hardly generalize to the burden of devices currently available. Recently, the need for considering users’ behaviour is highlighted [7]. A first attempt in this direction has been done by Kang et al. [4], where real data are used to build a personalized power model. They do not take into account power consumption specific to mobile device, which can behave differently even with a given user profile.

In this paper, we present MPower, the final implementation of the idea presented in [3], i.e., a system able to predict accurately an Android-based mobile device TTL, without any dedicated hardware tool or modification of the operating system. The model is focused on the hardware component power consumption, rather than users’ behavior, allowing comparison among devices. The whole architecture is based on a mobile application implementing a low power logger and a remote server, where the main computations are offloaded.

**System Architecture**

MPower relies on a system architecture based on two main components: an Android client logging application, running on the mobile device, and a remote server, for the data storage and the model computation. In Fig. 1 the information flow is presented. At first, the application on the mobile device gathers data related to energy consumption, e.g., battery level, CPU frequency, screen brightness, and sends them to a remote cloud server, responsible for the power model generation. Finally, the model is sent back to the device and the currently estimated TTL is shown by the MPower client.

![Figure 1: The MPower system architecture](image)

**Power Model Generation**

The power consumption model estimation phase is performed on data gathered by the MPower Android app:
no controlled environment experiments are required. MPower relies only on the everyday usage data, providing a flexible system, adaptive w.r.t. new devices and new OS versions and even able to handle battery degradation effects. Data gathered from the device may be grouped into two main categories: controllable actuators values and uncontrollable variables. The first group determines the device *configurations*: each assignment of actuators values determines device state. For instance, a configuration is WiFi OFF, 3G ON, bluetooth OFF, etc. Given a configuration, the uncontrollable variables, such as time elapsed, CPU usage, etc., are used as input for the TTL prediction. For each configuration, a power consumption model is computed, estimating an ARX model [5]. The power model is sent to the device as a look-up table, containing TTLs corresponding to each configuration. Thus, the mobile device does not have to perform any further computation, but a simple query on the look-up table.

**The Android application**

A preliminary version of MPower app is available on the Google Play store [2] and the complete version will be released in September¹. Currently, it logs the phone information and allows the user to visualize the phone usage statistics. In the final version it will provide also the TTL in the current configuration and will offer suggestion for achieving the desired TTL by changing configuration. Fig. 2(a) presents the application main page screenshot: the battery TTL for the current configuration is reported in the center of the screen, while on the lower part it is possible to access phone statistics (buttons “Reports” and “Charts”) and to allow the phone to refine the model if needed (button “Collecting new data”). The button “Set battery life” leads to the page in Fig. 2(b), where the user can change the configuration, to modify TTL. Moreover, the user can use the available filters to set constraints on functionalities the device must provide.

¹Further information available at [http://mpower.necst.it/](http://mpower.necst.it/)
Fig. 3 shows the third MPower feature: the visualization of the device usage statistics. Specifically, Fig. 3(a) reports a pie-chart with the different networks data usage distribution, while Fig. 3(b) visualize the overall network data usage and the power consumption.

MPower functionalities allows the user to keep the battery consumption under control, without having an excessive overhead in power consumption. In fact, since it runs continuously on the mobile device, a low impact on the battery is required. For this reason, to design a non-power-hungry application, the following aspects have been addressed during the application developing phase:

**sampling frequency** Sampling the device status every ten seconds allow the application not to be listed in the Android OS most battery draining apps;

**wake-lock** The application does not make use of any wake-lock, which would requires the device to leave the *deep-sleep mode* (an energy saving state);

**computation** The model estimation and adaptation is up to the remote server. To display the TTL, the only computation required by the device is a single query on a lookup table.

**data transfer** Samples are logged on small files, to avoid power consuming accesses to a single big file. These information are then compressed and encrypted, to create smaller packets, and are sent to the server only if the device is on charge.

**Conclusion**

We presented MPower, an Android app for power management on mobile devices. It is able to provide the current TTL of the phone, as well as to suggest configurations to save power and extend the battery life, by basing on a power consumption model. It also provides phone usage statistics, enhancing the user awareness on the mobile power consumption. The application logs data about the device, sends them to a server, which sends back the estimated power model, ultimately providing the TTL to the user. Thanks to this architecture the system is power-friendly, flexible w.r.t. new phones and adaptive to the specific user.

**References**