Algorithmic Optimizations for Energy-efficient Monitoring of Spatial Objects on Smartphones

Ulrich Bareth
Deutsche Telekom Laboratories, TU-Berlin
Service-centric Networking
ulrich.bareth@tu-berlin.de

Abstract: Location-based services are about to take the next step to proactive services which need continuous positioning for monitoring spatial objects. But so far that also results in severe battery drain in mobile devices due to deficient positioning APIs and the absence of energy-efficient positioning with several positioning methods. By using a hierarchical positioning concept this work provides a general algorithmic optimization to extend existing positioning APIs for energy-efficient positioning without diminishing accuracy. First prototypic implementations on Android handsets show a promising decrease in energy consumption.

1 Introduction

With so-called Location-based Social Communities on the rise (Foursquare, Gowalla, Friendticker, myTown, Yelp, Brightkite), location-based Services (LBSs) are gaining more and more momentum and are about to take the next step towards next generation LBSs, which are introduced in [KTLP06]. This class of services like geofence- or zone-based services as described in [MB09] will continuously keep track of a user's location and proactively notify her about potentially useful information in her vicinity like red light cameras, location-aware advertising or friends residing nearby. Therefore geographic events like entering a certain region of interest have to be monitored by tracking. To fulfill the needs of tracking, current mobile phones are equipped with several positioning methods that are based on the Global Positioning System (GPS), WiFi or Cell-Id, which mostly results in a high energy demand and thus quickly drain the device's battery. Because tracking with respect to energy-efficiency is not implemented in today's cell phone APIs, every application developer has to reinvent the wheel to minimize the device's energy consumption without neglecting the accuracy of position determination that is required by the application. This paper discusses already existing approaches and proposes a new concept for energy-efficient positioning by providing an API that significantly reduces the energy consumption for location tracking.

The remainder of this paper is structured as follows: Section 2 discusses commonly used positioning technologies, location APIs and related work in energy-efficient positioning for smartphones. In section 3 our approach is presented and evaluated qualitatively in section 4. The final section gives an outlook on future work and concludes the paper.

2 Positioning Technologies, APIs and Power Management

Nowadays smartphones like the iPhone or Android handsets not only utilize GPS for determining their location. Several methods are at hand that can be used for positioning such as cellular network positioning (Cell-Id) or WiFi positioning methods (WiFi) as described in [Küp05]. Recently a lot of research has been conducted in positioning, especially in the area of WiFi positioning like in PlaceLab and [CCLK05]. But also commercial implementations are avaliable today from Google and Skyhook Wireless. Those different positioning methods are relatively new to mobile devices and thus are not accessible in an optimal way through the provided application programmers interfaces (APIs) and usually are based on older APIs when tracking was not an addressed issue. Since those positioning methods have individual characteristics, they should be used in different ways to reduce energy-consumption. Hightower et al. investigated attributes of location systems for ubiquitous computing [HB01]:
Accuracy: the veracity of a determined position (deviation in meters)
Precision: degree of reproducability (percentage of fixes within that accuracy)
TTFF: time to first fix (time in seconds until location can be determined initially)
Power: energy consumption (in Watt seconds (Ws))
Availability: situations where positioning is limited

2.1 Characteristics of Positioning Mechanisms for Smartphones

GPS satellites continuously broadcast signals from space to GPS devices, which receive those signals to calculate their location. WiFi positioning uses wireless access point's MAC-addresses to uniquely identify WiFi hotspots whose positions have to be related to a reference position like a GPS fix and stored in a database. For positioning, the location of the MAC-addresses of the received hotspot signals is looked up in that database. Cell-Id positioning is similar to WiFi positioning, but uses the unique identifier of cell towers instead of a WiFi hotspot's MAC-address. The following table contains the characteristics of positioning mechanisms. Note that accuracy (Acc) and precision (Prec) are only coarse values to exemplify the order of magnitude of the attributes of today's smartphones [CSC+06]. The energy consumption is described as the amount of energy that is needed to obtain one position sample based on measurements from [WLA+09]. Figure 1 illustrates

Technology	Acc / Prec	Energy	TTFF	Limitations
A-GPS	10m / 95%	6.616Ws	15s	indoors and urban canyons
WiFi	50m / 90%	2.852Ws	3s	rural areas without WiFi coverage
Cell-Id	5000m / 65%	1.013Ws	3s	areas without cellular coverage

Table 1: Characteristics of positioning methods based on [WLA+09]

that the comparison of the introduced positioning methods can be seen as a pyramid. The narrow part of the pyramid, which is represented by GPS stands for high energy consump-

tion, high accuracy, but a long TTFF and limited outdoor availability. Cell-Id positioning at the bottom is vice versa. In the middle, WiFi positioning has moderate accuracy and energy consumption.

Figure 1: The positioning pyramid

2.2 Positioning APIs on Mobile Devices Platforms

In 2003 JSR-179 was released as an extension to J2ME and today is still the basis for many positioning APIs. It provides a generic API that retrieves the device's present physical location for Java applications and enables developers to write LBSs for resource limited devices. Though it does cope with separate location providers, it implements neither a concept for energy-efficienct tracking, nor does it support dynamic usage of multiple simultaneous location providers or proximity detection. The *Android* API extends JSR-179 by proximity detection, *RIM* provides a mechanism to keep the GPS receiver in a passive state to optimize TTFF. The *iPhone* API provides one single location interface that accepts a fixed accuracy but does not allow to address seperate location providers and *Windows Phone* provides none of the above mentioned mechanisms. In summary today's location APIs only allow for one single position fix, constant position updates or no position update at all. That results in a cumbersome usage, if you want to keep track of spatial objects nearby, because the relationship of many position fixes have to be checked with every spatial object.

2.3 Power Management for Positioning

A lot of research has been conducted in the area of power management in positioning. Kjærgaard's EnTracked platform [KLGT09] achieves optimizations on a Nokia N95 with its accelerometer, which turns GPS positioning off if the device is not moving. Farrel et al.[FCR07] reduced the amount of energy consumed by GPS and transmission of related data by reducing the number of position queries and updates to a server without neglecting certain accuracy of positioning. Zhuang et al. [ZKS10] use several power management mechanisms in Android as dynamic selection of location providers, but in a different scenario when some providers are not available. Also, they synchronize the location requests of parallel LBSs and deactivate positioning when the device is not moving as in [KLGT09]. Piggybacking reuses position data for several parallel LBSs. Although being

effective, those approaches are not directly applicable to our addressed topic of efficiently monitoring spatial objects, but some aspects of it can be integrated in future work.

3 Hierarchical Positioning for Monitoring Spatial Objects

Since positioning technologies on today's mobile phones have different characteristics concerning power consumption, there is huge potential to improve energy-efficiency while sustaining the required accuracy. The algorithmic optimizations of this work comprise a hierarchical positioning concept for tracking, which activates and deativates different positioning technologies on demand. Our optimization encapsulates a hierarchical concept into a tracking API, which provides simple functions to monitor spatial objects without caring about internal positioning details. Positioning methods are automatically disabled by default and only activated on demand. Figure 2 shows an example of how the hier-

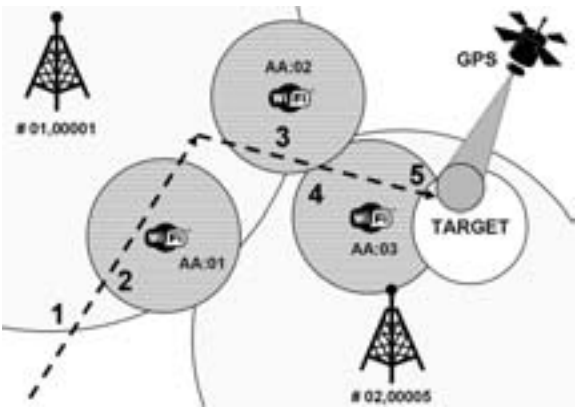

Figure 2: Illustrated functionality of the hierarchical positioning algorithm.

archical positioning algorithm works. In the scenario, a user wants to be notified when she enters the white circular target area, which could be the vicinity of her home. Slashed arrows describe the movement of the mobile device. The reception of mobile network provider's cells are illustrated by big circles with yellow vertical stripes, a base station symbol and a simplified Cell-Id. Smaller circles with horizontal orange stripes, the WiFi logo and a simplified MAC-address represent the coverage of WiFi access points. The red circle represents a GPS position fix. Numbers describe relevant steps for the positioning algorithm:

Step 1: The user's device detects that a new cell tower (#01,00001) is in its vicinity. Therefore the alogrithm determines by Cell-Id positioning whether the target area is inside that location but since it is completely outside cell coverage there is currently no need for other positioning methods.
Step 2: A WiFi hotspot (AA:01) is within reception range, but the device does not recog-

nize it since except Cell-Id all other positioning methods (including WiFi) are not in use to save energy. After the user took a turn right, the same situation occurs when entering the WiFi hotspot (AA:02).
Step 3: Again the device detects a new cell tower (#02,00005) . Now the algorithm checks the spatial relationships again and detects that the target now lies within the Cell-Id location. Therefore the next more accurate positioning method (WiFi) has to be activated to determine whether the user is already entering the target zone but it detects that the user is still outside the target area and the next more accurate positioning (GPS) method stays deactivated. WiFi positioning stays active to monitor when the user leaves the current WiFi coverage.
Step 4: As the user leaves its current WiFi hotspot (AA:02) and enters a new one (AA:03), WiFi positioning reveals that the user might potentially reside within the target area, because the WiFi zone coincides the target zone. Thus GPS positioning is activated to get more accurate location information, which shows that the user is still outside target area. Therefore GPS stays active.
Step 5: Finally, a GPS position fix lies within the target area and the desired notification is shown to the user.

The above description shows that WiFi and GPS, which consume much energy are deactivated most of the time and only activated on demand. The algorithm defines a general approach, which can be applied to any existing location API and integrate other positioning methods in a generic way. An easy-to-use extension for current positioning frameworks can be realized with the following two methods. *getCurrentLocation(int acc)* is used to immediately receive one location fix of a device, given the required accuracy in meters. *monitorRegion(Area region, int event, Listener monitor)* has a listener as callback interface for events that will be thrown when the provided areas, which define the required accuracy are being entered or left. The major problem that inhibits algorithmic optimizations is that all current mobile location APIs let the programmer decide what positioning method should be used. Our approach in contrast automatically determines the most energy-efficient method, which is still accurate enough for the application's current needs.

4 Evaluation

First implementations of a prototype have shown that algorithms for dynamically switching between different positioning methods have a tremendous potential for reducing power consumption in LBSs. In the prototypic implementation the uptime of an *HTC Nexus One* of nearly 6 hours in a naive implementation (GPS always active) could be increased to 15 hours in a GPS sleep mode (20 seconds sleep duration after each fix), reducing the energy consumption by 60%. Further evaluation will be conducted in a newly implemented LBS project at Deutsche Telekom Laboratories. During beta phase various users will be using that implementation and a naive implementation as reference to measure the power saving effects.

5 Conclusion and Outlook

Since this paper introduces early work in progress, a lot of implementation and evaluation still needs to be done to verify the feasibility of this general approach. Initially the different characterisics of the positioning mechanisms have to be compared under realistic conditions, since there is not much data available in scientific work. In the long term, energy-efficient tracking of a user's position should be implemented on OS level and integrated into the location API.

References

[CCLK05] Y.C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm. Accuracy characterization for metropolitan-scale Wi-Fi localization. In *Proceedings of the 3rd international conference on Mobile systems, applications, and services*, page 245. ACM, 2005.

[CSC+06] M. Chen, T. Sohn, D. Chmelev, D. Haehnel, J. Hightower, J. Hughes, A. LaMarca, F. Potter, I. Smith, and A. Varshavsky. Practical metropolitan-scale positioning for gsm phones. *UbiComp 2006: Ubiquitous Computing*, pages 225–242, 2006.

[FCR07] Tobias Farrell, Reynold Cheng, and Kurt Rothermel. Energy-Efficient Monitoring of Mobile Objects with Uncertainty-Aware Tolerances. In *IDEAS '07: Proceedings of the 11th International Database Engineering and Applications Symposium*, pages 129–140, Washington, DC, USA, 2007. IEEE Computer Society.

[HB01] J. Hightower and G. Borriello. Location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66, Aug 2001.

[KLGT09] Mikkel Baun Kjærgaard, Jakob Langdal, Torben Godsk, and Thomas Toftkjær. EnTracked: energy-efficient robust position tracking for mobile devices. In *MobiSys '09: Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 221–234, New York, NY, USA, 2009. ACM.

[KTLP06] A. Küpper, G. Treu, and C. Linnhoff-Popien. TraX: A device-centric middleware framework for location-based services. *IEEE Communications Magazine*, 44(9):114–120, Sept. 2006.

[Küp05] Axel Küpper. *Location-based services : Fundamentals and Operation*. John Wiley & Sons Ltd., 2005.

[MB09] J. Martens and U. Bareth. A declarative approach to a user-centric markup language for location-based services. In *Proceedings of the 6th International Conference on Mobile Technology, Application & Systems*, pages 1–7. ACM, 2009.

[WLA+09] Yi Wang, Jialiu Lin, Murali Annavaram, Quinn A. Jacobson, Jason Hong, Bhaskar Krishnamachari, and Norman Sadeh. A framework of energy efficient mobile sensing for automatic user state recognition. In *MobiSys '09: Proceedings of the 7th international conference on Mobile systems, applications, and services*, pages 179–192, New York, NY, USA, 2009. ACM.

[ZKS10] Zhenyun Zhuang, Kyu-Han Kim, and Jatinder Pal Singh. Improving Energy Efficiency of Location Sensing on Smartphones. In *Proceedings of the 8th international conference on Mobile systems, applications, and services*, pages 315–330, 2010.