What’s in the Apps for Context?  
Extending a Sensor for Studying App Usage to Informing Context-awareness

Abstract
Mobile phones became multi-purpose devices supporting their users with large variety of applications for various tasks. Not only the number of available applications is increasing, also the number of applications people are using on their devices is growing, as well as the amount of time people spent on their smartphones daily is getting bigger. In this workshop paper, we briefly describe our past work on understanding mobile application usage. We explain our research tool for measuring mobile application usage, called AppSensor, and discuss possibilities to exploit the information of mobile application usage to inform the reasoning about users’ contexts. We contribute our source code to the workshop for a discussion and prototyping of use cases leveraging the information of which application a user is currently using.

Author Keywords
Mobile applications; virtual sensor; context-awareness; app stores; studies; in the large; open source.

ACM Classification Keywords
H.5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous
Introduction

Smartphone applications (apps) have recently gained large momentum. The mobile ecosystem has made a rapid growth, and the number of available applications is steadily increasing. Also the number of downloads is growing rapidly, and it is anticipated to surpass 81.4 billion in 2013.1 Smartphones became multifunctional devices, which integrate a huge variety of services beyond pure communication capabilities. In addition to making phone calls, people can use their mobile phones for instance for navigation, mobile payment, making videos, taking photos or gaming and entertainment. The variety of activities that people use their smartphones for, and the number of installed applications is further increasing [4].

Our community has begun to study this mobile ecosystem to understand how people make use of all the applications that are available (cf. e.g. [1,3,7]). Further, following Want's "personal proxy" [8] line of thinking, the growing mobile ecosystem allows us to leverage application usage to learn about user's activities; e.g., if somebody is using a sports app like RunKeeper she is most likely doing her workout.

This paper summarizes studies that we conducted on mobile application usage in the large using a framework we developed, called AppSensor. Elaborating on that, we discuss opportunities for leveraging this tool for conducting research on and building services for context-awareness. With this paper, we want to accompany the source code2 of AppSensor that we plan to contribute to the workshop. Our aim is to contribute the AppSensor as a specific virtual sensor for integration into the AWARE framework, leveraging its open architecture.

Concept of AppSensor

We designed our framework along the lifecycle of mobile applications, as shown in Figure 1. The core idea is to determine which application a user is currently actively using on his smartphone. Therefore, the AppSensor infers which application is actively being used in terms of showing the user interface to the user, rather than which application is active as a software process from the operating systems point of view. As such, the sensor tells much more about the user and his current activity than about the internal state of the smartphone operating system.

Implementation of AppSensor

We implemented the concept of the AppSensor for the Android Platform. This decision was led by the platform's openness and its ability to access every piece of sensor-based data on the one hand (like latitude, longitude, altitude, speed and orientation). On the other hand, Android provides the possibility to get many information about the device itself, e.g. resolution and model name, and events related to an application's lifecycle on the device, e.g. installation, update and removal.

The core part of AppSensor is implemented as a background process running as a service. This service registers listeners for events that we want to capture within the system: usage of other applications (self-defined event), app installations, app removals, app updates, system boot, device power off, screen off, screen on. In addition, our implementation from time to time takes snapshots of the full list of applications that are installed, and determines which applications the

---

1 http://www.gartner.com/it/page.jsp?id=2153215
2 http://url.available.when.paper.gets.publish.ed
user can launch through an icon in his application launcher.

To summarize, our tool is implemented using many different Broadcast Receivers (provided by the operating system) to get notified about status changes relate to installed applications and hardware components, as well as the interaction flow of the user.

The application usage traces are collected at a configurable frequency, depending on the current use case. For tracing data without any loss, all collected pieces of information are immediately stored in a local mobile database. As soon as the framework detects that a connection to wireless LAN is available, the stored data will be sent to a central storage.

**Studies Done with AppSensor**

In a first phase of the development of the AppSensor we have used the framework for running large-scale studies for understanding the principles of mobile application usage and creating supportive functionalities, which we briefly summarize here to show capabilities of the framework.

*Understanding app usage at large-scale*

We deployed a first version of AppSensor to an application released on the Google Play Market [1]. We were able to collect a large dataset on mobile application usage of over 4,100 participants over more than four months. We found context-related patterns of application usage, e.g. when people are traveling they are more likely to use multimedia applications, and news applications are more popular in the morning.

*Multitasking between applications*

In another study we investigated the data collected through the AppSensor to look into mobile application interruptions [5]. We found that application interruptions happen only rarely, but if they happen they can have a significant impact on the application usage times, especially if interruptions on concurrent app usage are cause by incoming phone calls.

*Recommendation of mobile applications*

To support the discovery of mobile applications on app stores we built a recommender system leveraging the AppSensor in two ways: On the one hand it informed the recommender engines about which applications have been used in which contexts to provide recommendations on similar applications. And on the other hand we used the AppSensor for a usage-centric evaluation of the recommender engines [2].

*Prediction of Application Usage*

Most recently we also used the AppSensor as basis of an approach for the prediction of application usage to build a context-aware shortcut launcher of applications [6]. Leveraging the AppSensor and parts of the context information that it provides in addition, we built an adaptive app launcher menu and released it to the Google Play Store.

**Discussion: What’s in the Apps for Context?**

We think that in a second phase of our work with AppSensor we can go beyond exploiting the collected data for "only" analyzing application usage. Based on evidence from our studies and related work (cf. e.g. [2,3,7]) we know that application usage relates to user context. Now we argue that the information about which application a user is currently using is also of rich value for reasoning on a users context (in addition to the other sensors that are already taken into account). This is the inversion of the approach we used in Phase 1 to study application usage.
In the scenario sketched in Figure 2, the idea is to tell apart two different activities of users in a case where other sensors (e.g. location, time, acceleration, body activity) would provide same data. In this example one user is using a tourist application (left), and the other one is using a shopping list application (right). Based on this information provided by AppSensor an intelligent system might be able to tell apart the two activities of sightseeing and shopping nearby the mall.

In particular, this idea is motivated by our findings that application usage relates to context variables like location and time of the day. This gives rise to other systems using the approach we developed to take application usage as an input and reason on context.

**Conclusion and Future Work**

In this paper, we describe the concept of AppSensor and summarize the results of studies that we conducted by leveraging our framework. Further, we discuss how we think information on application usage can be used as a rich data source to inform context reasoning.

For the hands-on work at the workshop we would like to discuss an integration of the AppSensor into the AWARE framework by providing a context plugin. For future work, we plan to extend the AppSensor to build a platform for researchers, application developers and the scientific community to collect and make use of application usage data. We already started to build a cloud-based architecture based on Amazon Web Services and for analysis of the large-scale data (that we collect through apps we deployed to the Google Play Market). We are using the Apache Hadoop framework for statistical analysis.

**References**


