SAINT: A Scalable Sensing and Inference Toolkit

Mashfiqui Rabbi[1], Thiago Caetano[3], Jean Costa[1], Saeed Abdullah[1], Mi Zhang[2], Tanzeem Choudhury[1]
[1]*Cornell University*, [2]*Michigan State University*, [3]*University of Campinas*
ms2749@cornell.edu, thiagocaetano1991@gmail.com, jmd487@cornell.edu, sma249@cornell.edu, tkc28@cornell.edu, mizhang@msu.edu

1. INTRODUCTION

Past few years have witnessed a momentous rise of continuous mobile sensing systems. This continuous sensing and data processing capability opens the door to a wide range of novel mobile applications in context-aware computing [4], personal fitness tracking [2] and well-being monitoring. Given this vast array of possibilities, it is very common that multiple mobile applications running on the same mobile phone utilize the same set of sensors and have very similar data processing and inference procedures. For example, a physical fitness application and fall detection both need access to the accelerometer and extract similar motion-related features. But, since there is no unified framework, accelerometer access and feature computations are *redundantly duplicated* across these two applications. Furthermore, some mobile inference applications need access to multiple sensors and activity inference results. For instance, a sleep inference application needs access to activity recognition inference, ambient sound recognition and light sensors [1]. However, no current systems provides (1) an easy to use API to get activity recognition, ambient sound recognition etc. to aid sleep inference (2) provides a way to make sleep inference available to other developers who can build on the results. In this poster, we present SAINT that can address the above mentioned challenges. The objective of SAINT is two-fold: (1) to provide a *unified* framework to share sensor data and common data processing procedures across different mobile sensing applications; and (2) to provide a *extensible* and *scalable* solution where new capabilities can be developed that support reuse of *any* existing sensing and data processing capabilities.

2. THE SAINT ARCHIETECHTURE

SAINT realizes the unified scalable solution goal by posing the problem as a publish-subscribe [3] (or pub-sub) architecture. In a "pub-sub" system, *publishers* produce data which subscribers listen to. However, the *subscribers* number can vary and is unknown to the publishers. In order to support such communication, pub-sub introduces an intermediate "event bus" or "broker". Subscribers register for specific publisher data to the broker. On the other hand, publishers send data directly to the broker and the broker relays the information to registered subscribers.

In SAINT, producers are of two types: (1) sensing modules that record different sensor data and make them available. We call these sensing modules *sensor detectors*. e.g., a microphone detector publishes the audio data (2) Inference modules process the sensor data, perform inferences and make them available. We call these modules Signal Processing and Inference (SPI) Engines e.g., a voice recognition SPI will process audio and classify them as human voice or not. On the other hand, subscribers are applications that require sensing and inference data from the publishers. These subscribers can be a standalone application that uses SPI or sensor detector data. If multiple of such standalone applications want same publisher data then publisher just

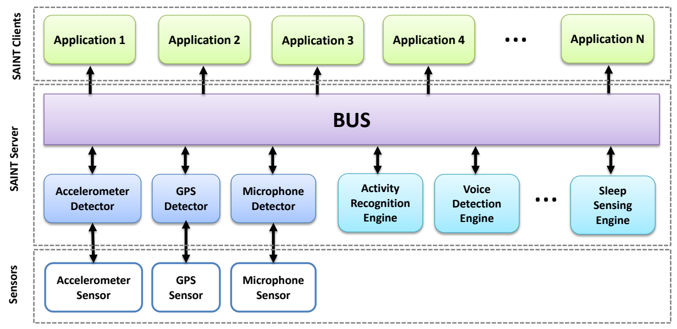

Figure 1: SAINT Architecture

produces the data once and broker relays the information to multiple subscribers, thereby reducing redundancy. Furthermore, sensor detectors or SPIs can also be subscribers that re-uses data from existing publishers to extend new capabilities. For example, a sleep SPI can use voice recognition, activity recognition and location to understand user movements and ambient noise levels to decide whether the user is sleeping.

3. ONGOING WORK

SAINT is a work under development. We already have a working implementation of SAINT that can be found in this link[1]. With current version, several sensing and inference capabilities can be accessed and extended with a few lines of code. In our future work, we will address the following challenges: (1) SAINT has centralized control to co-ordinate and schedule data delivery. Efficient management of these delivery (e.g., batching) can save battery. (2) we will implement access control protocol to manage who can access what sensing and inference capabilities. This will ensure better data privacy. (3) currently SAINT is implemented in Android. However, we received several suggestions for an iOS version of SAINT for higher adoption. We will implement SAINT for iOS in our future work.

4. REFERENCES

[1] Z. Chen, M. Lin, F. Chen, N. D. Lane, G. Cardone, R. Wang, T. Li, Y. Chen, T. Choudhury, and A. T. Campbell. Unobtrusive sleep monitoring using smartphones. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2013 7th International Conference on*, pages 145–152. IEEE, 2013.

[2] S. Consolvo, D. W. McDonald, T. Toscos, M. Y. Chen, J. Froehlich, B. Harrison, P. Klasnja, A. LaMarca, L. LeGrand, R. Libby, I. Smith, and J. A. Landay. Activity sensing in the wild: A field trial of ubifit garden. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1797–1806, New York, NY, USA, 2008. ACM.

[3] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec. The many faces of publish/subscribe. *ACM Computing Surveys (CSUR)*, 35(2):114–131, 2003.

[4] M. Mun, S. Reddy, K. Shilton, N. Yau, J. Burke, D. Estrin, M. Hansen, E. Howard, R. West, and P. Boda. Peir: the personal environmental impact report, as a platform for participatory sensing systems research. In *in Proc. ACM/USENIX Int. Conf. Mobile Systems, Applications, and Services (MobiSys) Krakow*, 2009.

[1]http://bit.ly/SAINT-HM. Name:hotmobileSAINT,Pass:GTAkKvs2