一般社団法人 電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report

CSSP: Cooperative Sensor-data Stream Protocol

Ryosuke OZAKI, Niwat THEPVILOJANAPONG §, Teemu LEPPÄNEN §§, Naofumi KITSUNEZAKI, and Yoshito TOBE

Department of Integrated Information Technology, College of Science and Technology, Aoyama Gakuin University
5-10-1 Fuchinobe, Chuo-ku, Sagamihara, Kanagawa, 252-5258 Japan
E-mail: {ryosuke.ozaki, yoshito-tobe}@rcl-aoyama.jp, kitsunezaki@it.aoyama.ac.jp

§ School of Engineering, Mie University

1577 Kurimamachiya-cho, Tsu City, Mie Prefecture, Japan 514-8507 Japan

E-mail: wat@net.info.mie-u.ac.jp

§§ Faculty of Technology, University of Oulu

Central University Administration, Pentti Kaiteran katu 1,

P.O. BOX 8000, FI-90014 Finland

E-mail: teemu.leppanen@ee.oulu.fi

Abstract In this paper we present the design and implementation of Cooperative Sensor-data Stream Protocol (CSSP) for smartphone-based participatory sensing applications. The objective of CSSP is scheduling the activities of sensing and transmitting data among clients to reduce the total energy consumption of clients. In CSSP, a server manages all the sessions between the server and clients and determines a transmission interval at each client. The CSSP client receives the information about the interval and transmits the data based on the notified interval. We have designed CSSP messages over HTTP and implemented client software on Android OS. Our preliminary experiments show that CSSP can contribute to the reduction of total energy consumption for the same sensing effect.

Keyword Networked Sensing, Mobile Sensing, Cooperation

1. Introduction

Due to the proliferation of smart phones, participatory sensing is being paid much attention to. One of the problems in participatory sensing is its random sampling and the incompleteness of sensing coverage. Often, more than necessary amount of sensing data is uploaded for the same location at the same time. This will result in ineffective energy consumption of smart phones.

To cope with the above problem, we design a protocol for cooperation among smart-phone clients called Cooperative Sensor-data Stream Protocol (CSSP).

In CSSP, we assume that clients do not directly communicate with each other. Rather, they only communicate with a server which has connections with each client. Thus the server control all the information about the locations and session time of the clients and determines a transmission interval of each client. The CSSP messages are built on HTTP and this will necessitate rate control at the application level.

We have designed CSSP messages over HTTP and implemented client software on Android OS. Our preliminary experiments show that CSSP can contribute to the reduction of total energy consumption for the same sensing effect.

The rest of the paper is organized as follows: Section 2 compares related work with our research. Sections 3 and 4 describe the design and implementation of CSSP, respectively. Section 5 shows the results of preliminary experiment. Section 6 concludes this work.

2. Previous Work

CenceMe [2] integrates sensing presence and social networks by capturing a user's current activity status and sharing such information in social network. By developing and evaluating a prototype on a Nokia N95 mobile phone, the energy consumption was studied at different sensing intervals, i.e., the user determined the sampling rate in advance and thereafter fixed it through-out the recording duration. BikeNet [1] allows cyclists to share information on themselves and the paths they traverse. Bicycles are equipped with a Nokia N80 mobile phone and Moteiv Tmote Invent motes and other necessary sensors. The sampling rate in the continuous sensing mode is set according to the cyclist's preference profile.

Energy Efficient Mobile Sensing System (EEMSS) [5]uses hierarchical sensor management strategy to recognize user states as well as to detect state transitions. The states are sensor assignment functional block to turn sensors on and off based on a user's current condition. By powering only a minimum set of sensors and using appropriate sensor duty cycles EEMSS significantly improves device battery life. Musolesi et al. [3] resent several techniques for uploading sensed information when connection is available. Uploading activity attempts to reach an acceptable trade-off in terms of ac-curacy and energy consumption given the requirements of the sensing systems. The server uses Markov model to predict the current state when fresh information is not present. The fresh information is sent if and only if the server information diverges from that currently calculated in real-time on the mobile phones. Aquiba [5] s a cooperative sensing approach where each user adjusts upload rate according to the number of nearby users. To minimize the upload rate, each user utilizes short-range radio to discover nearby users. Then sensing task is distributed equally among the users. This work differs from the previous work in that we do not assume inter-phone communications.

3. Design

In this section we describe the design of CSSP. The CSSP messages are designed as requests from a client and its response from a server.

Entities

In CSSP, two entities are defined: CSSP-client and CSSP-server.

CSSP-client: In many cases, global IP addresses are not assigned to smart phones in an Internet Service Provider. Therefore we allow a private IP address for a client. This leads to a protocol design in which a client initiates sending a message to a server as its communication counterpart. The CSSP-client also continuously senses data and transmits them to a CSSP-server. Unlike video or audio streams, the interval between two consecutive transmission ranges in the order of minute.

CSSP-server: It manages CSSP-client information and data and controls an interval of sensing and transmitting data. It corresponds to an incoming message from CSSP-client and responds to it.

Protocol Stack

CSSP is implemented over HTTP, because ports without well-known ports can be closed for security.

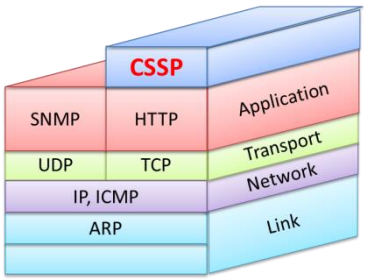

Figure 1 Protocol Stack

Messages

- CSSP-client

1. CheckIn

The CSSP-client notifies the server of current location and type of sensor list.

2. PutData

The CSSP-client notifies the server of sensor-data, date, and time.

3. CheckOut

The CSSP-client notifies the server of the end of sensing.

- CSSP-server

1. ACKforCheckIn

The CSSP-server notifies the client of client peculiar ID, type of sensor, an interval.

2. ACKforPutData

The CSSP-server notifies the client of latest interval.

3. ACKforCheckOut

The CSSP-server notifies the client of a message that tells your information has been deleted.

Message Format

Messages are represented by JSON.CSSP uses CSSP code in order to discriminate CSSP messages.

- CSSP-client

1. CheckIn

```
{ “csspcode” : CHECK_IN ,
 “latitude” : 35.5395 ,
 “longitude” : 139.4514 ,
 “sensortypes” : [1 , 2 , 3 , 4 , …]
}
```

2. PutData

```
{ “id” : 1,
 “date” : “2012-12-22” ,
 “time” : “12:30:00” ,
 “value” : 34.5 ,
“pre_times” : [ “2012-12-22_12:00:00” ,
“2012-12-22_11:30:00”] ,
 “pre_values” : [ 35.5 , 33.3 ]
}
```

3. CheckOut

```
{ “csspcode” : CHECK_OUT ,
“id” : 1
}
```

- CSSP-server

4. ACKforCheckIn

```
{ “id” : 1 ,
 “sensortype” : 3 ,
 “interval” : 30 ,
 “csspcode” : CHECK_IN
}
```

5. ACKforPutData

```
{ “csspcode” : PUT_DATA ,
 “interval” : 60
}
```

6. ACKforCheckOut

```
{ “csspcode” : CHECK_OUT ,
 “message” : “Thank you for your sensing.”
}
```

Algorithm of determining an interval of sensing and transmitting data

Equation (1) means an algorithm of determining an interval of sensing and transmitting data for CSSP-client. M shows the number of data needed every hour. N shows the number of CSSP-client.

$$(60 / M) * N \qquad (1)$$

4. Implementation

.

- CSSP-client

CSSP-client is implemented as Android application. This consists of CSSPclient and SensingService.

1. CSSPclient

It communicates with CSSPserver and control SensingService.

2. SensingService

It communicates with CSSPserver, schedules sensing time, and senses data.

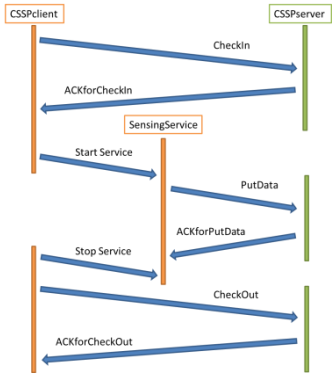

Figure 2 CSSP Messages

- CSSP-server

CSSP-server is implemented as HTTP server written by Java. This consists of CSSPserver and ConnectionThread.

1. CSSPserver

It creates ConnectionThread, if it receive a socket from CSSP-client.

2. ConnectionThread

It analysis a socket and processes data according to every message.

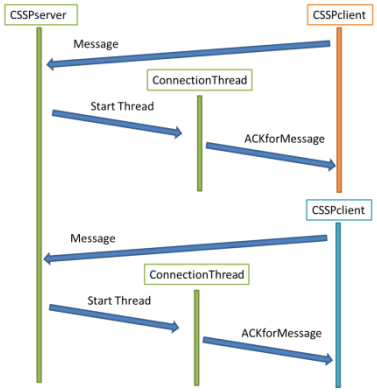

Figure 3 Connection Threads

5. Experiment

We conducted an experiment using CSSP-implemented Android mobile phones for noise-level sensing. This experiment was done for two hours at one specified area. As the number of clients increases, the transmission interval changes between 1, 5, 10, 30 minutes. One session for the same interval lasted 30 minutes and the residual quantity of battery and battery consumption rates were measured. Figures 4 and 5 show that changing transmission interval caused by the CSSP control was effective in reducing the total energy consumption.

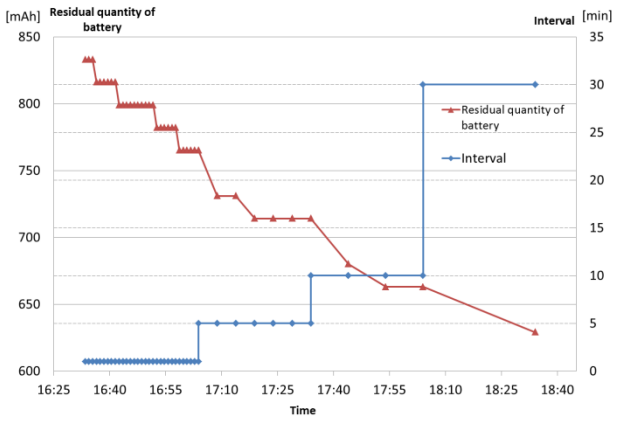

Figure 4 Residual quantity of battery

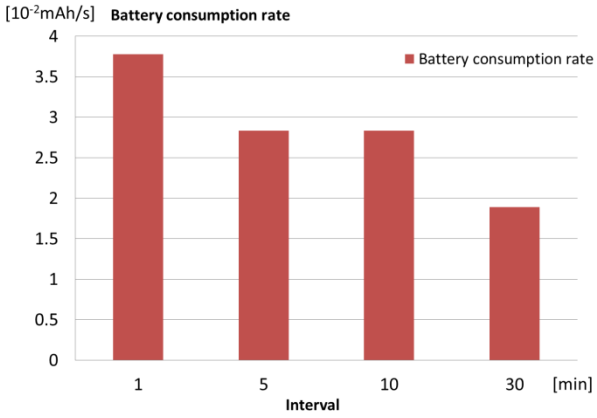

Figure 5 Battery consumption rate

6. Conclusion

This paper describes the design and implementation of CSSP. Although CSSP is beneficial when users move around with mobile phones, we have not designed a case for mobility. This remains for our future work.

References

[1] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell. The BikeNet mobile sensing system for cyclist experience mapping. In Proceedings of the 5th International Conference on Embedded Networked Sensor Systems (SenSys 2007), pages 87–101, Sydney, Australia, Nov. 2007.

[2] E. Miluzzo, N.D. Lane, K. Fodor, R. Peterson, H. Lu, M. Musolesi, S.B. Eisenman, X. Zheng, and A.T. Campbell, “Sensing meets mobile social networks: the design, implementation and evaluation of the CenceMe application,'' Proceedings of the 6th International Conference on Embedded Networked Sensor Systems (SenSys 2008), Raleigh, North Carolina, USA, pp.337-350, Nov. 2008.

[3] M. Musolesi, M. Piraccini, K. Fodor, A. Corradi, and A. T. Campbell. Supporting energy-efficient uploading strategies for continuous sensing applications on mobile phones. In Proceedings of the 8th International Conference on Pervasive Computing (Pervasive 2010), pages 355–372, Helsinki, Finland, May 2010.

[4] Niwat Thepvilojanapong, Shin'ichi Konomi, and Yoshito Tobe, "A Study of Cooperative Human Probes in Urban Sensing Environments," IEICE Trans. Comm,Vol.E93-B, No.11, pp. 2868 - 2878, Nov. 2010.

[5] Y. Wang, J. Lin, M. Annavaram, Q. A. Jacobson, J. Hong, B. Krishnamachari, and N. Sadeh. A framework of energy efficient mobile sensing for automatic user state recognition. In Proceedings of the 7th Annual International Conference on Mobile Systems, Applications, and

[6] Services (MobiSys 2009), pages 179–192, Krakow, Poland, June 2009.