International Journal of Electrical & Electronics Research (IJEER)
Volume 3 issue 3, Pages 44-49, September 2015, ISSN: 2347-470X

Design of Middleware and Software Embedded Development Kit For Area Based Distributed Mobile Cache System

Victoria Satuluri[1], Ratna Babu Yellamati[2]
[1,2]Department of ECE, Vignan's Lara institute of technology and science, Vadlamudi, Guntur-522213
[1]Victoria.s895@gmail.com, [2]ratna2k5@yahoo.com

Abstract — *Nowadays a new area-based distributed mobile cache system and a sustainable distributed Geo-cast technology enables data caching temporarily in a designated local area using short-range communication technologies such as Bluetooth and Wi-Fi. This system doesn't need servers and infrastructure network. To enhance the short-range communications as a time-offset communication in a physical world the caching capability is needed. Due to this advantage it is sure to adopt a new location-based communication services. The proposed work is frames a developed middleware for a widespread embedded OS and a software development kit with an open application programming interface. The proposed middleware is expected to prompt the third-party developers to produce various applications using this platform and provided with some pseudo-code. And also explained the methods for using this platform by an implemented geo-location social game.*

Index Terms: Application programming interface, Area-based distributed mobile cache, Software Development Kit, Application programming interface.

I. INTRODUCTION

Nowadays a wireless communication plays a vital role in many daily usages [1]. The short-range wireless technologies have been recently developed in the form of WiFi and Bluetooth gives a solution for implementation of embedded systems with information access. Mostly, the advanced communication technologies provide a instantaneous channel between unknown passengers on a street and enables the information exchange between one to another. Thus, the communication technology is known as opportunistic communication, which will help to form a temporary local-area-related mobile network instantaneously in the real environment. So these types of instantaneous networks are used in many innovative applications nowadays. These applications can be real-time short message exchange application and an individual profile exchange application. In rapid expanding applications the real-time social battle game on a portable game device is used. Due to the utilization of this technology it has gained explosive popularity and produced a large business market in real time environment. Therefore, these services are effective and enjoyable only for a person belongs to same area at the same time due to the communication occurs in a real-time environment. If the communication with time-offset inside a prescribed area were achievable then it would give a solution for a new communication style promoting the integration of a huge number of completely new applications. Thus area-based caching capability encourages a sustainable social network construction and its communication across time is expected to evolve. Among one research technology an Abiding Geo-cast [5] provides a time-stable location oriented information delivery to all nodes that are inside a target area within a certain intervals. The author's group earlier proposed a new area-based distributed mobile cache platform to cache the data temporarily in a designated local area [6]. The proposed new area-based mobile cache platform is formed by distributed multiple terminals. And it is maintained by sharing and relaying cache data among the multiple terminals that pass in or near a local area network. Effectively this system can operate any areas such as in crowded city areas or in large exhibition centers.

This platform should be easy to use, so that many third-party applications and services can easily adopt. Due to latest circumstances of mobile applications, especially for widespread embedded environments the proposed platform should be wrapped as a middleware library and should be distributed to the public accompanied with an easy-to-use software development kit (SDK). Thus the authors newly developed a middleware cache system to serve this purpose. This middleware operates by the latest widespread embedded operation system. And also the middleware has a developer-friendly application programming interface (API) to support development of various applications.

The rest of this paper is organized as follows. Section II describes the summary of related works. Section III explains details of the existing system. Section IV describes the implemented middleware in detail. In Section V presented the introduction of three applications using the developed SDK and in addition to details of the social game application section VI describes the performance evolution.

II. RELATED WORK

A. Geocast and Abiding Geocast

A Geocast is proposed for an area-based advertisements applications and position based message sharing [7] applications. The Geocast routing protocols are classified into two types [8] such as a flooding-based protocol and a route setup based protocol [5]. While flooding-based protocol transmits a message in flooding manner to all terminals inside a predefined area. This type is categorized as Location Based Multicast (LBM) and GeoGRID [11]. Simultaneously, route-setup-based protocol organizes a delivery route beforehand and delivers a message to the terminal inside the prescribed area. GeoNode approach [12] and a GeoTORA approach [13] are categorized as this type.

An Abiding Geocast is proposed for a sustainable cache which is a time-stable geocast delivered to all nodes that are inside a designated area within a certain intervals [5]. This cannot guarantee reliability of message delivery. This has been proposed before the Area-Based Distributed Mobile Cache System and aimed to provide temporal caching capability for non-real and time-offset communication functionality [6]. This proposal requires no such route or network construction. To reduce network overload among terminals it has been proposed.

B. Middleware and SDK deployment for embedded consumer devices

To make the porting of complex software easier and more efficient for handheld platforms, the portable software development kit was proposed. Particularly for those with proprietary operating systems such as the popular portable game terminal was developed. The EcoSpire is one of the hardware and software platform for various wireless sensing applications [18]. This hardware kit consists of two types of self-contained and expandable sensor nodes, also a multipurpose base station and programmer, with cascadable charger and sensor modules. While the software kit consists of an integrated development environment (IDE) for application programming and radio frequency (RF) debugging utilities on the host computer with a lightweight operating system which support for data logging and remote firmware reprogramming. Additionally, a simple toolkit was proposed to simplify the process of building robust and extensible wireless sensor networks with environmental sensor networks [19]. It includes a programming language called the sensor web language (SWL) and a graphical user interface for writing SWL programs also a compiler for generating a complete stack of deployable sensor network code.

III. AREA-BASED DISTRIBUTED MOBILE CACHE SYSTEM

This section briefly describes the area-based distributed mobile cache system with its system diagram and basic operation. The diagram and operation are logical. The practical classifications have been described in the following section.

A. Basic Concept

This cache system is accomplished by transferring and sharing information across multiple mobile devices via short-range wireless communication and thereby producing an area-based 'virtual cache' through collaborative access of devices while avoiding a network infrastructure. As the basic concept is described in Fig. 1. Basically, every terminal moves directly around in a real physical environment. As soon as the end mobile terminal decides to cache the data in a present area, then it attempts to cache data attached to its target cache area information: landmark information (Fig. 1(a)). Afterwards, these cache data are transferred to neighboring terminals (b)(c). Thus the receiving terminals decides whether the received data should be cached inside the current area or not, then attempt to cache it temporarily and if necessary it re-share with other terminals. Diversely, the cached data are deleted outside the area (d). Every terminal can be a relay terminal candidate for any cache data in any area. Thus, it might carry multiple cache data in multiple cache areas (e). In this cached data might be used if applications on these devices want to use the data.

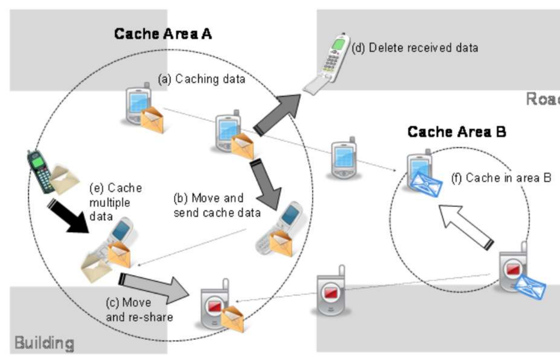


Fig. 1. Area-based distributed mobile cache system [6].

B. Basic Operations

As shown in the figure 2 the collaborative mobile cache module located inside of each terminal operates as a middleware which are separated from third-party applications. From Fig. 2(a) data are first cached in the Application Data Database according to a request from an application. And the Metadata are attached with the cache data which are its application id, the current timestamp, its data type and the target cache area information. The area and location information are always tackled in the Control Data Database (b) from the Location Controller using the Location Sensor. Relay timing is determined along with a terminal to share against each cache datum (c). If more than one cache datum is to be relayed then one packed datum created in the Relay Data Packetizer (d) is relayed to another terminal via the Transmitter Controller (e). As soon as the terminal receives relayed cache data to be relayed via the Receiver Controller (f) it decides whether each component relayed datum should be cached into the

Application Data Database or not by parsing the packed relay data. And each component relay datum is cached as independent different cache data and is notified to the corresponding application (g). Certainly the application can retrieve active data from the Application Data Database (h).

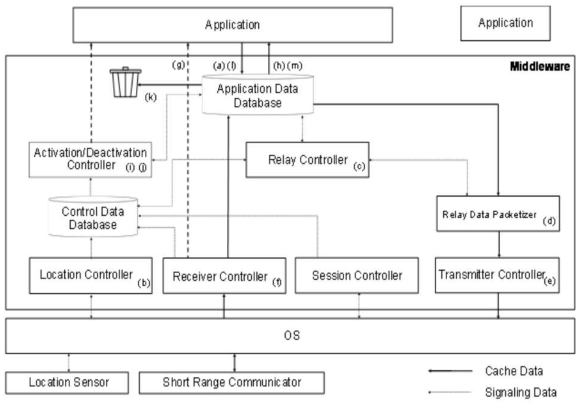


Fig. 2. System diagram and operational flow.

IV. MIDDLEWARE IMPLEMENTATION

The proposed middleware was developed for a widespread embedded operating system. Middleware terminals can easily form the distributed cache platform. To implement various applications easily by third-party developers, an open SDK including an open API was also employed to provide various controls for the developers and it is provided to the public as library files. The present middleware uses Bluetooth technology for direct communication between each terminal. Global Positioning System (GPS) is also used for measurement of terminal positions.

A. Implementation Architecture

The middleware is implemented by an Objective-C language native module, which is located between third-party applications and an operating system. The Bluetooth framework and GPS framework are used for communication and position management. The Bluetooth framework is used to detect other devices and controls establishment of devices, termination of channel, and status management of communication sessions. And it also provides data receiving and transmitting functionalities. The position of the terminal is managed by GPS framework according to its movement. The two frameworks are controlled by developed middleware via those proper APIs. Thus the middleware uses a newly developed API for communication with third-party applications. This API has two types: one is for applications to control the processes inside the middleware and other is to hook the processes from the middleware. Hence the latter API is so-called *callback API*. From the figure 3 it is designated as a *Delegate API*.

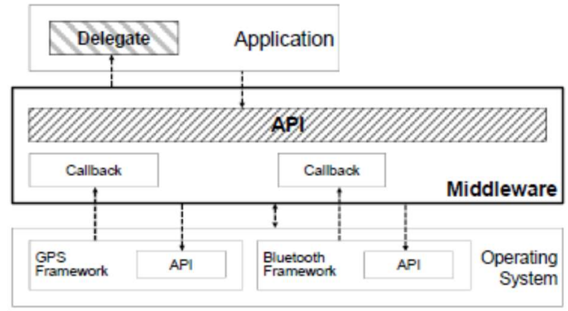


Fig. 3. Middleware, API and frameworks.

B. Middleware Process Implementation

The implementation details are explained below in Figure 4. Three types of middleware processes are explained.

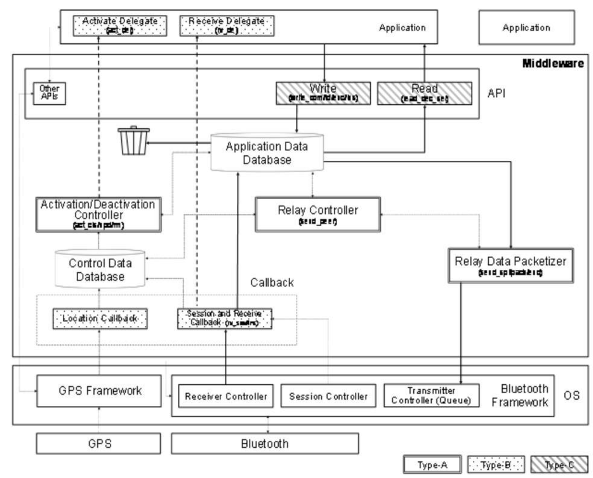
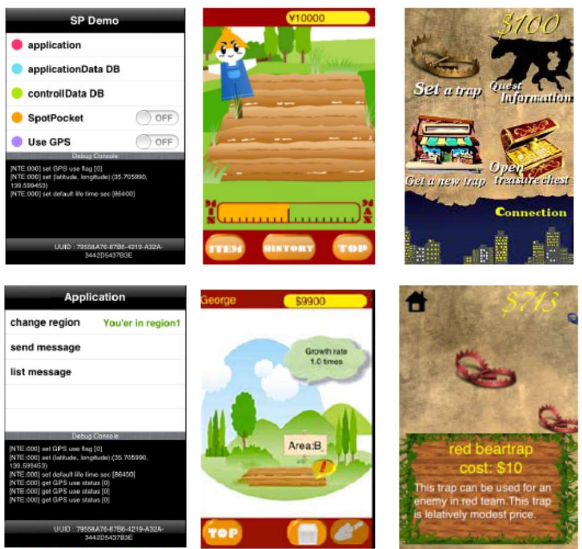


Fig. 4. Implemented middleware diagram

The first process is a timer process for data activation–deactivation and relay data transmission (Type-A). In this process the data packets are transferred to be relayed into an internal data queue inside the Bluetooth Framework. Second process is callback processes from the two frameworks (Type-B). When the status changes of surrounding terminals or communication session activates this process. This process also includes the registration of one's own terminal position into the control database when the GPS framework calls back. The last process is application-triggering processes via the API (Type-C). These processes include, data writing and reading into/from the Application Data Database inside the middleware.

V. IMPLEMENTED APPLICATIONS

Three applications were used to demonstrate how SDK is useful. First is a verification test application that helps application developers to explain the SDK behavior. This application provides developers various user functions to write/read application data to/from the middleware, to simulate device location change, to review cached data and to check detectable devices in proximity. The Screenshots are of the games are shown in Fig. 5(a). And the other two applications are geo-location social games. This makes users to enjoy games with passengers who walk through designated areas when traveling to school or work. The figure 5(b) is for users to grow vegetables with other people in an asynchronous manner and the vegetable fields are shared with people. And they unite to water the vegetables and destroy insects by fertilizing. Figure 5(c) describes the user challenge by virtually setting traps in the physical area.



(a) Verification Test (b) Vegetable Growing (c) Trap Game

Fig. 5. Applications using deployed SDK.

VI. PERFORMANCE EVALUATION

The performance evaluation of the developed middleware was described in this section. The processing time in each process was measured to identify bottleneck processes. Mostly, the data processing time is increase is measured according to the change of data size as well as the number of data. A data-transmitting terminal and a data-receiving terminal were prepared for evaluation of the performance. All databases in both terminals were refreshed into empty at the beginning of the experiment. For evaluation a specific application was employed. The target processes to be studied were located between the writing process and the reading process. Using **write** method the data is written from the evaluation application on the data-transmitting terminal, the data are activated, and packed into the relay data format and transmitted to the data-receiving terminal. As soon as the receiving terminal receives the data, using **read all** method the evaluation application reads the received data. All of these processes were measured. And the embedded terminal had 1 [GHz] CPU, and 512 [MB] memory. Bluetooth interface version was 2.1.

TABLE III
PROCESSING TIME VS. NUMBER OF DATA [S] (10^{-3})

Number	1	20	40	60	80	100
write_com	6.549	6.476	6.329	6.326	6.357	6.707
write_id	2.277	45.729	91.737	138.360	184.189	230.342
write_enc	1.539	30.224	59.479	88.810	119.105	149.018
write_ins	**22.721**	**462.782**	**949.822**	**1422.948**	**1906.736**	**2391.406**
act_rm	5.173	5.597	5.748	5.974	6.098	6.418
act_cls	**12.190**	**17.783**	**24.005**	**30.183**	**36.544**	**43.070**
act_upd	0.018	0.019	0.019	0.019	0.019	0.021
snd_peer	**10.505**	**17.586**	**22.828**	**28.843**	**35.798**	**40.488**
snd_spl	0.050	0.121	0.195	0.275	0.374	0.437
snd_pac	0.148	0.599	1.110	1.695	2.057	2.526
snd_enq	0.410	0.565	1.440	1.302	1.540	1.867
rcv_sav	**9.897**	**196.515**	**394.612**	**596.769**	**800.649**	**1021.909**
rcv_ins	**25.173**	**452.860**	**899.781**	**1353.359**	**1820.258**	**2335.649**
rcv_del	1.517	22.849	45.836	68.448	91.374	114.689
read_sel	6.438	11.796	17.317	22.833	28.414	34.850
read_dec	1.547	25.717	51.683	77.152	104.141	131.339

TABLE IV
PROCESSING TIME VS. DATA SIZE [S] (10^{-3})

Size [bytes]	100	2000	4000	6000	8000	10000
write_com	5.195	5.256	5.291	5.290	5.271	5.259
write_id	2.032	2.034	2.050	2.076	2.035	2.053
write_enc	1.067	1.156	1.255	1.377	1.487	1.637
write_ins	**21.991**	**29.424**	**37.191**	**43.620**	**52.091**	**53.660**
act_rm	4.379	4.643	4.619	4.216	4.223	4.340
act_cls	**9.375**	**9.530**	**9.746**	**9.709**	**9.862**	**9.938**
act_upd	0.020	0.160	0.017	0.017	0.016	0.016
snd_peer	**9.737**	**9.995**	**9.954**	**10.187**	**10.135**	**10.184**
snd_spl	0.038	0.039	0.039	0.040	0.041	0.039
snd_pac	0.141	0.155	0.169	0.185	0.200	0.207
snd_enq	0.411	0.385	0.593	0.651	0.814	1.017
rcv_sav	**15.722**	**15.574**	**15.605**	**15.625**	**15.475**	**15.485**
rcv_ins	**21.473**	**25.987**	**36.640**	**44.099**	**50.816**	**46.270**
rcv_del	1.595	1.681	1.802	1.899	2.021	2.124
read_sel	5.276	5.414	5.480	5.603	5.634	5.773
read_dec	1.090	1.168	1.244	1.329	1.392	1.459

TABLE III and TABLE IV respectively depict the processing times of the all processors when the data size and the number of data are changed. For comparison of two cases the data size was changed from 100 bytes to 10000 bytes for the former case. Simultaneously, the latter case altered the number of the data with 100 bytes from 1 to 100. From these results the whole processing time was occupied by data insertion processes in data writing and data reading and increased proportionally to the size change and the number of data. Further, comparing two results under the similar situations which are in the first row in the both tables, obtained the almost similar results.

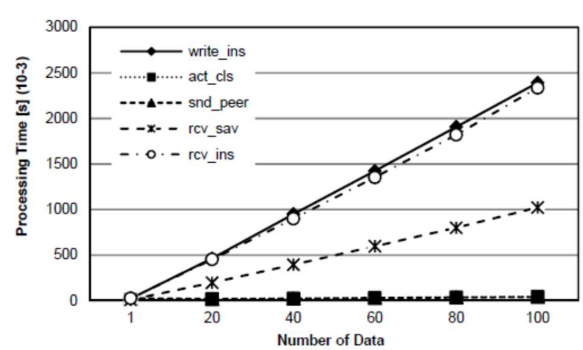


Fig. 6. Processing Time vs. Number of Data with 100 Bytes.

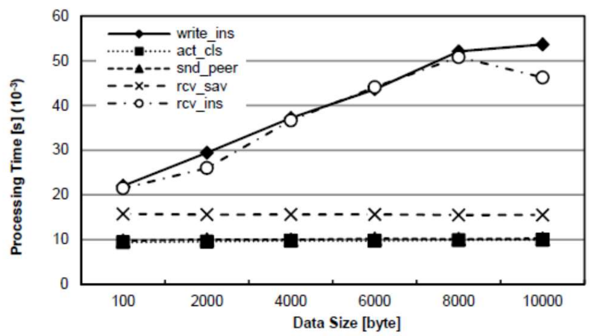


Fig. 7. Processing Time vs. Data size

CONCLUSION

The proposed described a design of middleware implementation of the distributed mobile cache system especially for a widespread embedded operating system and a software development kit with an open application programming interface. In this paper a detailed explanation of the developed middleware and the open application programming interface was studied. The procedures for using this platform were explained using an implemented geo-location social game. The employed SDK which is flexible and easy to use has great potential to drive new applications such as a geo-local message exchange and targeted advertisement. It is also targeted for real-time social games applications because they illustrate rapidly expanding applications. Furthermore, they have achieved explosive popularity and a large business market.

REFERENCES

[1] P. Tamarit, C.T. Calafate, J.C. Cano, and P. Manzoni, “BlueFriend:Using Bluetooth technology for mobile social networking,” in Proc. Of Sixth Annual International Conference on Mobile and UbiquitousSystems: Computing, Networking and Services (Mobiquitous 2009), 2009.

[2] A. Mumtaz, and F.T. Shah, “Games over Bluetooth,” IEEE International Conference on Computer Systems and Applications, pp. 1167–1170, 2006.

[3] L. Wang, E. De Vial, L. Tokarchuk, Y. Wang, and A. Ma, “Blue Danger: Live Action Gaming Over Bluetooth,” in Proc. of Sixth IEEE International Conference of Consumer Communications and Networking Conference (CCNC 2009), pp. 1287–1291, 2009.

[4] A.S.Y. Lai, and A.J. Beaumont, “Mobile Bluetooth-Based Multi-player Game Development in Ubiquitous Computing,” Journal of Computational Information Systems, vol. 6, no. 14, pp. 4617–4625, 2010.

[5] C. Maihöfer, T. Leinmüller, and E. Schoch, “Abiding Geocast: Time-Stable Geocast for Ad Hoc Networks,” In Proc. of the Second ACM International Workshop on Vehicular Ad Hoc Networks (VANET 2005), pp. 20–29, 2005.

[6] Hiromi Narimatsu, Hiroyuki Kasai, and Ryoichi Shinkuma, “Area-based Collaborative Distributed Cache System using Consumer Electronics Mobile Device,” IEEE Transactions on Consumer Electronics, vol. 57, no. 2, pp. 564–573, 2011.

[7] C. Maihöfer, and R. Eberhardt, “Time-stable Geocast for Ad Hoc Networks and its Application with Virtual Warning Signs,” Elsevier Special issue of Computer Communications, vol. 27, no. 11, pp. 1065–1075, 2004.

[8] M. Mauve, J. Widmer, and H. Hartenstein, “A survey on position-based routing in mobile ad hoc networks,” IEEE Network, vol. 15, no. 6, pp. 30–39, 2001.

[9] Y.-B. Ko, and N.H. Vaidya, “Geocasting in mobile ad hoc networks:Location-based multicast algorithms,” in Proc. of the Second Workshop on Mobile Computing Systems and Applications (WMCSA 99), pp. 101- 110, 1999.

[10] Y.-B. Ko, and N.H. Vaidya, “Flooding-based geocasting protocols for mobile ad hoc networks,” Mobile Networks and Applications, vol. 7, no. 6, pp. 471–480, 2002.

[11] W.-H. Liao, Y.-C. Tseng, K.-L. Lo, and J.-P. Sheu, “GeoGRID: A geocasting protocol for mobile ad hoc networks based on GRID,” Journal of Internet Technology, vol. 1, no. 2, pp. 23–32, 2000.

[12] T. Imielinski, and J. Navas, “GPS-based addressing and routing,” Internet Engineering Task Force, Network Working Group, Request for Comments, RFC 2009, 1996.

[13] Y.-B. Ko, and N.H. Vaidya, “GeoTORA: A protocol for geocasting in mobile ad hoc networks,” in Proc. of the Eighth International Conference on Network Protocols (ICNP), pp. 240–250, 2000.

[14] H. Mineno, Y. Kawashima, S. Ishihara, and T. Mizuno, “Collaboration Mechanism for Mobile IP based Link Aggregation System,” Journal of Information Processing Society of Japan (IPSJ), vol. 47, no. 7, pp. 2224–2235, 2006.

[15] G. Tsuchida, and S. Ishihara, “Replica Arrangement for Location Dependent Data in Consideration of Network Partition in Ad Hoc Networks,” International Journal of Communication Networks and Distributed Systems (IJCNDS), vol. 2, no. 4, 2009.

[16] Hong Lu, Nicholas D. Lane, Shane B. Eisenman, and Andrew T. Campbell, “Bubble-sensing: Binding sensing tasks to the physical world,” Pervasive and Mobile Computing. vol. 6, no. 1, 2010.

[17] Filippo Battaglia, Giancarlo Iannizzotto, and Francesco La Rosa, “An Open and Portable Software Development Kit for Handheld Devices with Proprietary Operating Systems,” IEEE Transactions on Consumer Electronics, vol. 55, no. 4, pp. 2436–2444, 2009.

[18] Chien-Ying Chen, Yu-Ting Chen, Yi-Hsuan Tu, Shun-Yao Yang, and P.H. Chou, "EcoSpire: An Application Development Kit for an Ultra-
Compact Wireless Sensing System," IEEE Embedded Systems Letters, vol. 1, no. 3, pp. 65–68, 2009.

[19] J.-P. Arp, and B. G. Nickerson, "A user friendly toolkit for building robust environmental sensor networks," in Proc. of the Fifth Annual
Conference on Communication Networks and Services Research, CNSR '07, pp. 76–84, 2007.

[20] H. Narimatsu, and H. Kasai, "Selective Data Deactivation Mechanism in Sustainable Area-based Cache for Mobile Social Networks,"ACM/Springer Journal Mobile Networks and Applications, vol. 17, no. 6, pp. 820–830, 2012.