Improving Computational Efficiency for Personalized Medical Applications in Mobile Cloud Computing Environment

George Mathew, Zoran Obradovic
Center for Data Analytics and Biomedical Informatics
Temple University
Philadelphia, PA, USA
{George.Mathew, Zoran.Obradovic}@temple.edu

***Abstract*— Mobile computing and cloud services are two technologies that have gained momentum in recent times. The proliferation of mobile computing devices and network connectivity has made it an attractive platform for delivering personalized services in many business domains including healthcare. Personalized health and wellness mobile applications have computational and data requirements that are necessitated by the localized processing needs of the application. The on-demand provisioning capability and elasticity of cloud services combined with the local processing capability of mobile devices can provide an ecosystem for pervasive access to health information. In this study we explore some of the specifics of these health and wellness applications. We introduce promotion algorithm as a mechanism to efficiently process data points locally by mobile devices. This algorithm can take advantage of the local processing power of smartphones and help reduce communication costs between mobile endpoint and cloud-based long-term data services. Experiments were performed using an Android smartphone for real time data acquisition of more than 10 million data points and a Linux server in a private cloud over 4G network simulating a health service. Results showed that the proposed algorithm could help preserve battery life by a factor of 10 and reduce data communication time by a factor of 20 as a result of utilizing local computation on a mobile device.**

Keywords - medical informatics; mobile cloud computing; personlized wellness.

I. INTRODUCTION

The architecture and design of personalized applications are influenced by social computing platform experiences of the masses. This is necessitated by the familiarity of common user interfaces in the social context. There are two service areas where these influences are well pronounced. First one is mobile computing and the second one is cloud services. Both these technologies have been commercially successful and have been widely researched. Needless to say, both technologies are past their emerging stages and are mature. Users at large are used to mobile phone based computing environment as well as social cloud services.

The proliferation of mobile computing devices has introduced some popular features and capabilities that can be taken advantage of, in the personal health and wellness applications space. The affordances offered by mobile computing devices are commonplace and so applications can make use of these. For example, a mobile application can be set with different alert tones for different events. Texting apps have the capability to set different delivery tones for different friends. In a similar vein, a medical app can use distinct alert tones to signal different severity levels of a medical condition to the patient. Timer capabilities, alarm capabilities, time zone change etc. are all features that can be taken advantage of. Hardware-dependent operations like swiping, zooming and pinching are operations natural to the mobile computing platform. The accessible from anywhere mantra of mobile devices are attractive for personal apps.

Mobile applications can be deployed in two ways. One is as a web-based application using html5. The other is as a native application. The correspondence of mobile application deployment scenarios with traditional desktop application deployment formats is shown in Figure 1.

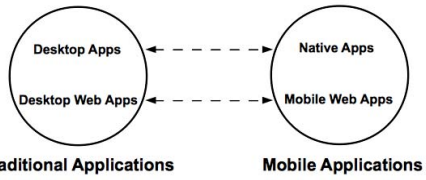

Figure 1. Deployment Formats of Traditional and Mobile apps

Mobile web applications run on the server-side, but the device needs continuous Internet connection. The desktop web applications and mobile web applications runs using a web browser within the graphical user interface. However, the desktop applications are installed from a media and the native mobile apps are usually downloaded from the app store. The native apps run locally and needs Internet connection only when data needs to be pushed to a cloud service. Native applications can take advantage of many hardware affordances. But these are device-dependent. Our focus is on the native applications for personal health and wellness.

Mobile phone users are used to downloading applications from the app store. Practically, these applications do not require any training for the end user. In general, users are comfortable figuring it out. Downloads

978-0-7695-5089-3/13 $26.00 © 2013 IEEE
DOI 10.1109/ICHI.2013.83

from Apple's app store and Google play store average more than a billion per month. This trend clearly shows the potential for a personal health application to be embraced by mobile phone users. The app distribution has two models. One is through a single source (as in the case of iPhone apps). The other is through any store (as in the case of Android apps). The two store scenarios present two different models that a personal healthcare app provider should choose between. In 2012, iPhones and Android phones accounted for more than 85% of the mobile market place. Even though mobile tablet devices have functionalities very similar to mobile smartphones, in this paper, our focus is strictly on smartphones – due to the pocket size small physical form factor. This small form factor is a convenience for patients who need to be interacting with a mobile application continuously.

In spite of these advantages, there are some recent incidents that are disheartening. The possibility of viruses on mobile computing devices was known for almost a decade now [1]. Though Apple has a tough gatekeeper policy for the App Store, a hacker was able to get a rogue application into the App Store [2]. In a recent publication by IDG, it was mentioned that the number of malicious and high-risk apps for Android is expected to triple from about 350,000 in 2012 to more than a million in 2013 [3]. This is an alarming trend for smartphones as an infected device can disturb the working of a medical wellness application that is critically important for a patient.

Another challenge related to smartphones is the co-mingling of functionalities. A smartphone can be used as a cell phone, video recorder/player, camera/photo gallery, music player, texting device, etc. They are also used to running multiple applications concurrently. Consequently, multiple processes vie for computing resources of the device. This has to be taken into consideration when a personal health app is designed.

On a different technology spectrum, enabling consumerization of IT services, cloud computing [4] have been gaining momentum. Elasticity of resources is one big advantage of cloud environment. However, co-tenancy of customers has raised questions about data security and data pilfering. Ironically, co-tenancy helps the provider to bring down service costs. One of the major issues with cloud service is the non-standard proprietary entry points. This makes it difficult to switch cloud provider in a seamless fashion. One of the attractive cloud services is data storage. However, the API's provided by one vendor is non-portable to another vendor and so customers gets locked to a cloud service provider. Data protection while hosting data with a provider should be seriously considered. This involves channel protection as well safeguarding data at rest. Channel protection can be accomplished via different symmetric and asymmetric encryption mechanisms [5]. Information Lifecycle Management calls for mechanisms similar to timed data shredding [6] so as not to leave data trails. For production data at rest, on the fly encryption techniques exist. Appliances that serve as encryption gateways (example – Voltage [7]) are available. Ideally, homomorphic encryption [8] is able to handle encryption of data in the cloud.

The general layout of a multi-user computing services environment is shown in Figure 2.

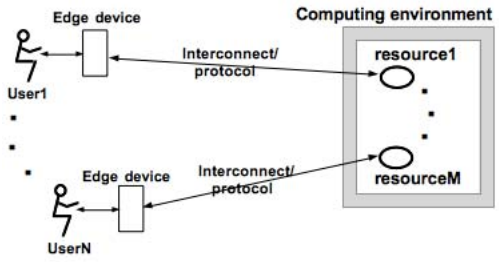

Figure 2. Reference layout of a multi-user computing environment

In the 1960's, when multi-user computing environments were introduced, users connected to the services from dumb terminals using serial protocol. In today's mobile cloud computing environment, users connect to services or resources via mobile computing devices using TCP/IP protocol. Thus the architectural representation in Figure 2 is valid from both current and historical perspectives. Consequently it may be argued that "many cloud computing security problems are not in fact new, but often will still require new solutions in terms of specific mechanisms" [9]. For example, the data path from the mobile device to the service is through a provider of user's choice. In a corporate environment, the data path from a computing device to a server is within the confines of the corporate network and so less susceptible to eves dropping by an outsider. In the mobile scenario, the data path need not be within the corporate boundary. This is illustrated in Figure 3.

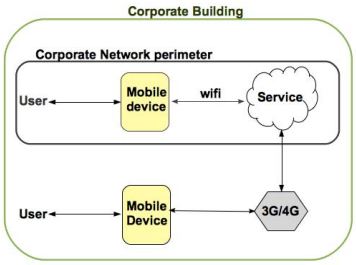

Figure 3. Mobile device connections to cloud services

As shown in the top inside box of Figure 3, the data path for a user connecting to a local service is within the corporate network. But, the access through a cell provider is outside the corporate network boundary (as shown in bottom half of Figure 3). Thus it is necessary to protect the data channel when public network is used. Since our focus is on personal health for the general public, the data path is almost always through a public network.

Mobile computing has the following advantages in personal healthcare domain: access to test results, emergency response and personalized monitoring (both off-band and online) [10,11] using an app. The mobile devices have multi-tasking operating systems. This makes it attractive for component-based applications. In spite of all these advantages, the mobile platform has its limitations. Smartphones have two channels – one for phone and one for data. Data bandwidth subscription dictates how much can be transferred without overcharge. When designing a data intensive health application that needs to communicate to a cloud service, the limit on the subscribed data bandwidth should be taken into consideration.

Table I. Hardware profiles of some common smartphones

Make	Model	Memory	Processor	Storage
Apple	iPhone5	1 Gb	Dual core 1.3 GHz	16-64 Gb
Samsung	Galaxy S III	1 Gb	Quad core 1.4 GHz	32 Gb
htc	8X	1 Gb	Dual core 1.5 GHz	16 Gb

Table 1 shows the hardware capabilities of some popular phones on the market. Most mobile devices have a graphics co-processor so as to support the very consumer-focused nature of the device for photos and videos. This hardware feature is advantageous for image-dependent patient care applications. The storage is used for applications, application-specific logs, data files, music files and photo/video files. Due to this multi-purpose use of the local storage and the personal preferences for applications, there is a wide variation in the memory and disk usage patterns. We collected the memory and disk usage from a random sampling of 20 users (10 iPhones and 10 Androids). Results are summarized in Figures 4 and 5.

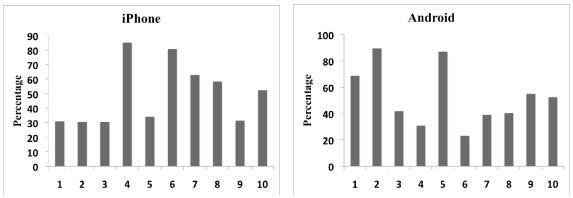

Figure 4. Memory usage of 20 random smartphones

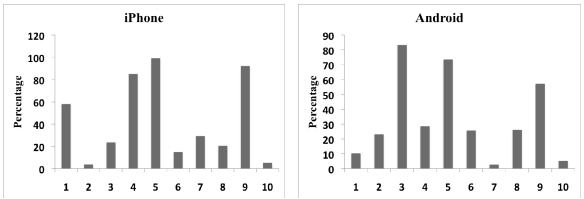

Figure 5. Disk usage of 20 random smartphones

As can be seen from Figures 4 and 5, the usage pattern for memory and disk is quite arbitrary between random users. Also, note that in desktop operating systems similar to unix/linux variants, if need be, a file system partition can be reserved with a predetermined amount of space for an application. Since on mobile devices it is only single multi-purpose storage, in the current systems, such reservation is not possible. It is also advantageous to process locally and send representative data to a central station – which is in the cloud. The cloud infrastructure could be public or private depending on the nature of the application. In this study, we are targeting native mobile apps that need local computational cycles and cloud data services.

II. Related Work

There are many health and wellness mobile applications geared towards educating the public on specific topics. For example, mobile apps related to prevention and care of HIV exists in the app stores [12]. @HealthCloud [13] is an example of a mobile app that requires data transfer between smartphone and cloud. This application was developed to present DICOM images on an Android phone using cloud data services from Amazon S3. In general, more data transfer between mobile phone and cloud service consumes more battery. Since battery life is a key resource that needs to be preserved, many studies have been done to prolong battery life. A model for battery life estimation [14] itself has been proposed. Cyber foraging [15], selective data reception [16] and Mobile proxy [17] are techniques suggested for efficient battery consumption. A stand-by method [18] for dual-mode mobile phones has been proposed to save power. Our focus is on applications that need to process data. Survey study has shown that [19] many users are not aware of the battery limitations on mobile phones and use applications or settings in non-efficient ways. Another study [20] has shown that each user's behavior and device features affect the resource utilization on the mobile phone.

III. Personalized Health and Wellness Apps

As mentioned previously, we target native mobile apps on smartphones. The small form factor of smartphones is a convenience for patients to carry around when they have sensors attached to their body that has to interact with a mobile application continuously through wireless feed. The NFC (Near Field Communications) [21] capability of a smartphone makes it possible to realize patient interactions with the sensors. In the following sections, we refer to native applications for smartphones simply as mobile apps.

There are many applications for personalized medicine that needs local processing and data communication to a cloud server. These are in vitro diagnostics (IVD) patient care applications. IVD helps real-time testing of patient, providing quality data and avoiding unnecessary trips to the clinic. This laboratory automation also helps document measurements that need to be taken at times even when the

clinic is closed. For example, Continuous Glucose Monitoring System (CGMS) is an FDA-approved device that can record blood sugar levels read from a sensor inserted under the skin of the patient's abdomen [22]. Currently, the data in the CGMS has to be downloaded to the lab computer manually after 3 days when the patient visits the lab. If the CGMS is a smartphone application that can interact with the lab computer, transactions will be real-time and medical decisions can be made earlier.

These IVD applications have the general format of needing local computation and sending results to a cloud service (either private or public). Such personal applications have sensors/devices attached to the patient's body or feed results to a mobile device from an implanted device. In these scenarios, the mobile phone is acting as a real-time data acquisition agent on behalf of the sensor or device. The general structure of this configuration is shown in Figure 6.

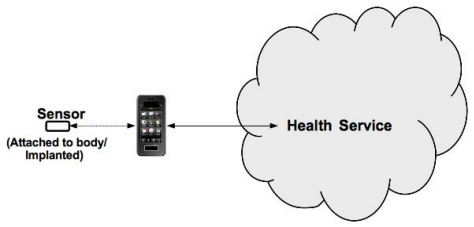

Figure 6. General Structure of a mobile-cloud wellness app

For example, in a published telemonitoring study of heart patients, single-lead ECG recordings (Selfcheck ECG PMP4, CardGuard, Israel) were sent wirelessly to a Blackberry smartphone via Bluetooth and then transmitted to the data repository at the hospital (private cloud) [23]. This example illustrates the above structure.

Based on the architecture for mobile cloud application shown in Figure 4, once the local computing on the mobile phone takes place, results can be pushed to the cloud service in one of two ways:

1. using a specialized protocol
2. using general Web Services

We do not delve into details here, as these are decisions to be made based on software design.

Due to the real-time nature of the IVD applications, a large amount of data will be generated. Instead of sending all data points to the cloud clearinghouse, it may be possible to do local decisions based on data within predefined time intervals. Sending large amounts of data over the carrier network can negatively impact battery power. In situations where a patient's medical condition is normal, there is probably no need to send data. Since smartphones have reasonably good computing power, a signature for the current interval can be computed and compared to the prior signatures. This is the idea behind the Promotion algorithm that is more formally outlined in the next section.

IV. PROMOTION ALGORITHM

This algorithm combines finer resolution time snapshots into a coarser resolution for an attribute measured over various times. Let there be k hierarchies of resolutions, starting with r_1 as the finest resolution and r_k as the coarsest. The algorithm should be able to handle arbitrary starting resolution and ending resolution within a window. A ***window*** corresponds to a set of data points whose time resolutions are completely within r_i time frame. The ***adjusted attribute value (aav)*** for a window will be a single calculated value representative for r_i time frame. The number of data points in the window for resolution r_i is represented as $count_i$.

For example, if k = 4, r_1 could be 10 minutes, r_2 could be 60 minutes (1 hr), r_3 could be 24 hours (1 day) and r_4 could be 1-week in resolution. Data points could be collected at each minute interval. This means $count_1$ is 10, $count_2$ is 6, $count_3$ is 24 and $count_4$ is 7.

The algorithm starts with an attribute of resolution r_x (from 1st data point) and keep on moving the data cursor, adding attributes to a bin (or basket) for the window, until an attribute beyond the current window is encountered. During this pass, when $count_x$ points accumulate in the bin, an adjusted attribute value (aav) is calculated as a representative value for the attributes in the current bin (as a collapsed value). Then this aav is promoted to the next bin corresponding to r_{x+1}. Once $count_{x+1}$ values accumulate in bin for r_{x+1}, the aav for this bin is calculated and promoted to the bin corresponding to r_{x+2} and so on until the promotion ends with r_k. The aav for the bin w.r.t. r_x is a representative value for the window corresponding to r_x. After the aav for a bin is calculated and promoted to the next bin, the entries in the bin are cleared. The formal version of the algorithm is given below:

Let r_1, r_2, ..., r_k be monotonically increasing units of resolution for time factors of interest for a temporal attribute u. Each r_s is a multiple of r_{s-1}. Assume that the time resolution t_i associated with each data point d_i is always known. The resolution of this value t_i can be any one of r_1, r_2, ..., r_k. This means the value t_i can be in one of these units. Another assumption is that the data points are sorted sequentially in time.

Promotion algorithm

```
variables: cursor, bin_resolution,
           binr1, binr2, …. , binrk
begin algorithm
   initialize binr1, binr2, …. , binrk to be empty
   while more data points are available
      keep a cursor on the next available data point
      bin_resolution = resolution of current point
      add current di attribute to bin_bin_resolution
      do
         while bin_bin_resolution is full
            compute adjusted attribute value
               (aav) for bin_bin_resolution
```

add aav to $bin_{bin\_resolution->next}$
empty $bin_{bin\_resolution}$
$bin_{bin\_resolution}$ = $bin_{bin\_resolution->next}$
end while
end do
end while
end algorithm

Illustration:

Assume $3*r_1 = r_2$ and $3*r_2 = r_3$. Here, r_1 is the finest and r_3 is the coarsest of resolutions. (r_1 is data captures at 20 minutes interval and the representatives are to be calculated for 3 hour intervals.) Consider the following 9 data points (in sequence) captured at resolution r_1.

0.50, 0.55, 0.57, 0.55, 0.54, 0.52, 0.60, 0.59, 0.52

Initially the three bins corresponding to r_1, r_2 and r_3 are [] , [], []. i.e, they are all empty. Once [0.5, 0.55, 0.57] bin is generated, the aav (in this case, we use mean for illustration) is 0.54. This aav will be promoted to the next bin for r_2. Now, the bins are [], [0.54], []. And so on. Here are the sequences:

bin for r_1	bin for r_2	bin for r_3
[0.5, 0.55, 0.57]	[]	[]
[]	[0.54]	[]
[0.55, 0.54, 0.52]	[0.54]	[]
[]	[0.54, 0.54]	[]
[0.6, 0.59, 0.52]	[0.54, 0.54]	[]
[]	[0.54, 0.54, 0.57]	[]
[]	[]	[0.55]

Hence the final aav is 0.55.

The aav can be calculated based on an algorithm suitable for generating a representative value for the bin. In the above example, aav for bin_{r1} could be mean, while aav for bin_{r2} could be mode and the aav for bin_{r3} can be harmonic mean.

V. EXPERIMENTS

We implemented the promotion algorithm using a Raspberrypi as the sensor, an Android smartphone for real time data acquisition and a Linux server in a private cloud simulating the Health service. All client and server software were written in Java. Raspberrypi was run using "Soft-float Debian wheezy" linux kernel so that we could install Embedded Java Virtual Machine on it. The server software for Android phone was written using Android SDK on Eclipse IDE. The Raspberrypi sent data points to the Android phone in realtime. From the Android phone, representative data points were sent to a virtual Linux server in the private cloud. A service was deployed on the Linux server to receive the data points. These components are shown in Figure 7.

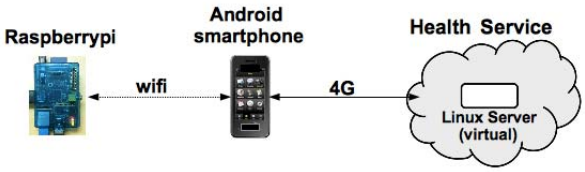

Figure 7. Components in the Implementation of Promotion Algorithm

The details of the Operating systems and Java Virtual Machines (JVM) are given in Table 2.

Table II. Versions of OS's and JVM's

Device	Operating System	JVM
Rapsberrypi	Debian wheezy Linux 3.1.9	Embedded JRE ver. 7
HTC	Android v 4.2.2	Dalvik 1.7
vmware virtual server	Linux CentOS 6	OpenJDK SE ver. 6

The Raspberrypi client generated random values (between 0 and 1) continuously and sent to the Android server. The promotion algorithm was implemented using three resolutions: r1 was set at 10 data points (finest resolution), r2 was set at 6 and r3 was set at 5 (coarsest resolution). At r1 resolution, the aav used was mean. At r2 level, the aav was maximum value from the 6 r1 aav's. For r3, aav was defined as range; i.e., max – min. We ran four experiments. In the first experiment, all points were sent to the Linux server from Android. In the second experiment, r1 was computed locally and sent to server. In the third experiment, r1 and r2 were computed locally and r2 was sent to the server. In the fourth experiment, all values r1, r2 and r3 were computed locally on Android and r3 was sent to the server. We tested the data transfer from the Android device to the private cloud over a 4G network. The battery discharge in different scenarios is given in Table 3.

Table III. Battery discharge over 4G

Smartphone computation	Cloud computation	Battery use	Time
<All points send to cloud>	aav1, aav2 & aav3	9%	18 mins
aav1	aav1 & aav2	2%	9 mins
aav1 & aav2	aav3	<1%	63 secs
aav1, aav2 & aav3	<none>	<<1%	54 secs

In each experiment, 10.8 million data points were generated at resolution r1. As seen from Table 3, the local computations on the smartphone resulted in faster results and substantially small battery power usage. The algorithm can be used to send an initial burst of data points within a window. If sufficient number of data points is sent in the highest resolution, a match for known medical conditions can be tested at the cloud service. Once the match is known, then an action can be taken on the remaining data points. It is also possible to use the highest resolution data points to establish a baseline profile. Once normal stage is reached, it is only necessary to send the periodic signatures at a lower resolution. The decision to process locally or in the cloud any one of the aav's at any resolution can be made based on the resource availability on the smartphone.

VI. CONCLUSION

We outlined the role of mobile computing devices and cloud services in the context of personalized healthcare and wellness applications. Various security issues relevant to the environment were elaborated. The focus was on smartphone apps that served as real time data acquisition systems from sensors/devices attached to patient's body. We introduced promotion algorithm as a mechanism to utilize the local computing power of smartphones and to efficiently use data channel. Experimental results show that the algorithm can help preserve battery life by utilizing local computation. This is useful for personal health and wellness mobile applications as battery life is an important factor.

ACKNOWLEDGMENTS

We are thankful to the volunteers who shared disk usage and memory usage statistics on their mobile phones.

REFERENCES

[1] D. Dagon, T. Martin, and T. Starner, "Mobile phones as computing devices: the viruses are coming!", IEEE Pervasive Computing, Oct – Dec, 2004. Vol 3, Issue 4, pp. 11-15.

[2] Apple lets malware in App Store, http://nakedsecurity.sophos.com/2011/11/08/apples-app-store-security-compromised/ Accessed April 2013

[3] Security and Mobile Device Management, http://resources.idgenterprise.com/original/AST-0082232_marble_byodsecyurity_v5.pdf Accessed April 2013

[4] M. Creeger, "Cloud Computing: An Overview", Distributed Computing, Vol. 7, No. 5, June 2009.

[5] G. Mathew, and X. Du, "Secruing Multi-tiered Web Applications", Proceedings of 2010 IEEE International Conference on Wirelss Communications, Networking and Information Security, Beijing, China, June 2010, pp. 505-509.

[6] M. Paul, and A. Saxena, "Proof of Erasibility for Ensuring Comprehensive Data Deletion in Cloud Computing", Communications in Computer and Information Science, Springer, Vol. 89, pp. 340 – 348. Proceedings of the Third International Conference CNSA 2010, Chennai, India, July 2010.

[7] Voltage key-based encryption appliance for the cloud http://voltage.com

[8] C. Gentry, "Fully Homomorphic Encryption Using Ideal Lattices", Proceedings of the 41st Annual ACM Symposium on Theory of Computing, Bethesda, MD, USA. pp. 169-178, May – June 2009.

[9] Y. Chen, V. Paxson, and R. H. Katz, "What's New About Cloud Computing Security?", Electrical Engineering and Computer Sciences, University of California at Berkeley, Technical Report No. UCB/EECS-2010-5. http://www.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-5.html Accessed April 2013.

[10] U. Varshney, "Pervasive Healthcare", IEEE Computer Magazine vol. 36, no. 12, 2003, pp. 138-140.

[11] S. Koch, M. Hägglund, I. Scandurra, and D. Moström, "Towards a virtual health record for mobile home care of elderly citizens", presented in MEDINFO 2004, Amsterdam, 2004.

[12] K. E. Muessig, E. C. Pike, S. LeGrand, L. B. Hightow-Weidman, "Mobile Phone Applications for the Care and Prevention of HIV and Other Sexually Transmitted Diseases: A Review", Journal of Medical Internet Research 2013; 15(1):e1. doi: 10.2196/jmir.2301.

[13] C. Doukas, T. Plikas, and I. Maglogiannis, "Mobile Healthcare Information Mangement Utilizing Cloud Computing and Android OS", Proceedings of 32nd Annual International Conference of the IEEE EMBS, Buenos Aires, Argentina, August 2010. pp. 1037-1040.

[14] T. D. Panigrahi, C. Chiasserini, S. Dey, R. Rao, A. Raghunathan, and K. Lahiri, "Battery life estimation of mobile embedded systems", Proceedings of 14th International Conference on VLSI design, 2001. pp. 57-63.

[15] J. Parkkila, "Improving battery life and performance of mobile device with cyber foraging", Proceedings of 2011 IEEE 22nd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Sep 2011, pp. 91-95.

[16] M. W. Kim, D. G. Yun, and S. G. Choi, "Battery Life Extension Method using Selective Data Reception on Smartphone", Proceedings of 2012 International Conference on Information Networking, Feb 2012, Bali, pp. 468-471.

[17] G. D. Mandyam, "Improving battery life for wireless web services through the use of a mobile proxy", Proceedings of 2010 IEEE 21st International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), Sep 2010, pp. 2763-2768.

[18] N. Imai, and K. Yoshihara, "A Power-saving standby method to extend battery life in dual-mode cell phones", In Proceedings of 2012 IEEE Consumer Communications and Networking Conference, Las Vegas, NV, Jan 2012, pp. 224-229.

[19] A. Rahmati, A. Qian, and L. Zhong, "Understaning Human-battery interaction on mobile phones", In Proceedings of the 9th International Conference on Human Computer interaction with Mobile Devices and Services", 2007. pp. 265-272.

[20] N. Vallina-Rodriguez, P. Hui, J. Crowcroft, and A. Rice, "Exhausting Battery Statistics", Proceedings of the second ACM SIGCOMM Workshop on Networking, Systems and Applications on Mobile Handheld, August 2010, New Delhi, India, pp. 9-14.

[21] J. Ylinen, M. Koskela, L. Iso-Anttila, P. Loula, "Near Field Communication Network Services", Proceedings of the 3rd International Conference on Digital Society, Cancun, Mexico, Feb 2009, pp. 89-93.

[22] Diabetes and Continuous Glucose Monitoring, http://diabetes.webmd.com/continuous-glucose-monitoring Accessed April 2013.

[23] E. Seto, K. J. Leonard, J. A. Cafazzo, J. Barnsley, C. Masino, and H. J. Ross, "Mobile Phone-Based Telemonitoring for Heart Failure Management: A Randomized Controlled Test", Journal of Medical Internet Research 2012;14(1):e31. Doi:10.2196/jmir.1909.