Context Awareness over Transient Clouds

Andrea Sciarrone, Igor Bisio and Fabio Lavagetto
University of Genoa, DITEN Department
Via All'Opera Pia 13, 16145, Genoa, Italy
Email: {andrea.sciarrone, igor.bisio, fabio.lavagetto}@unige.it

Terrence Penner and Mina Guirguis
Texas State University
San Marcos, TX 78666.
Email: {t_p68, msg}@txstate.edu

***Abstract*—The exponential increase in the number and types of mobile devices, along with their ever-growing sets of capabilities, have enabled the development of new architectures that aim to harness such heterogeneity. Transient Clouds (TCs) are examples of mobile clouds which are created on-the-fly by the devices present in an environment to share their physical resources (e.g., CPU, memory, network) and would disappear as the nodes leave the network. They enable a device to go beyond its own physical limitations through utilizing the capabilities offered by nearby devices over an ad-hoc network. In this paper we present a Transient Context-Aware Cloud (TCAC) in which the nodes of the network care more about providing/learning higher level functionalities rather than lower level capabilities. We make the case for such an architecture in scenarios where it is not feasible for all the nodes to compute the context due to privacy, energy, and delay constraints rather than an unreachable network. We present a prototype implementation of our architecture over Android smartphones connected via WiFi along with the performance metrics (power/energy consumption and accuracy) to show the benefits of context awareness in TCs.**

I. Introduction

Motivation: Despite the recent technological advances in mobile devices, a single device has many limitations in terms of the amount of computation that can be done locally and the extent to which power is consumed. This has prompted a lot of research efforts in the areas of cloud computing and mobile cloud computing in which computations are offloaded to the cloud [1], [2]. In some scenarios, however, the cloud is ill-suited to carry out the computation due to its lack of context – something that only the devices present in the environment can capture. It would be costly to have all the devices provide the cloud with that context. Moreover, delay and privacy constraints exacerbate such issues. Consider the case in which a computation is needed for a real-time context query: by the time the cloud service is invoked, the context has already changed. Similarly, performing the computation on the cloud (where auditing and monitoring tools are more prevalent) may not provide some users with the desired degree of privacy.

Understanding the local context is important in many applications. For example, in a health monitoring service setup it is important recognize certain activities (e.g., Is the person moving or not? Does this activity pattern resemble a known one? Does he/she sound distressed?, etc...) or lack of activity, and raise an alarm if the recognized activity does not match the expected one. The context should be tied to the time of day and the known patterns for that individual. Similar examples abound in other scenarios as well. As such, all the relevant information pertaining to the context must be considered – including the users and the applications themselves [3]. Often such information spans more than one device due to the inherent difference in capabilities supported by each device. Fusing such information for context awareness is our main motivation behind this work.

Clearly, context awareness requires some computation and communication among the devices present. The transient cloud model allows for devices to rely on their collaborative efforts to provide computational services without access to the infrastructure [4]. To that end, this paper presents an architecture that fuses a context awareness algorithm (see [5]) with transient clouds (see [4]). Rather than focus on sharing physical resources such as the CPU, memory, storage, and network, we focus on sharing higher level functionalities that are driven by the context. We argue that in many scenarios, devices are more interested in the context rather than the exact computation to be performed on a remote device. Furthermore, our proposed architecture does not require a connection to the infrastructure which may not always be available (e.g., disaster impacted sites and monitoring in remote areas) or usable in some situations (e.g., high traffic load that congest the network links).

Illustrative examples: Consider a major traffic congestion incident in which all the drivers – through their smartphones – are trying to compute alternative routes. Despite that all drivers are going towards different destinations, getting out of this congested region start with a much smaller set of routes. It would be worthwhile for only a few of the devices to download the new maps, compute the alternate route, and share the result (i.e., the context) with other drivers, rather than have all the drivers try to download the maps – creating network congestion – and perform almost the same exact computation themselves.

In another example, consider a private meeting among a group of users in which the goal is to infer and share high-level information about the meeting such as how many people were attending, who was speaking, and for how long. With context-aware services, devices can detect that the meeting is ongoing and, based on their capabilities (and also proximity to certain users), some can start recording while other devices can run feature selection and identification to extract who spoke and for how long. Such information can be shared with all the users after the meeting has concluded (e.g., meeting minutes). This can be realized without sending information over the traditional cloud service, since a high degree of privacy may be necessary.

One of the main contributions of this work is merging the concept of Transient Clouds with Context Awareness and demonstrating the usefulness of this approach through

978-1-4799-5952-5/15/$31.00 ©2015 IEEE

experimental evaluation with a heterogeneous set of devices in realistic applications. Furthermore, we focus on energy consumption for measuring the effectiveness of our platform.

Paper organization: In Section II we describe various pieces of work related to our proposed platform. In Section III we describe our proposed platform in detail. We present our experimental results in Section IV and we conclude the paper in Section V with a summary.

II. Related Work

The continuous development of hardware sensors on mobile devices has enabled the devices to pervasively recognize the social context of their environment. Consequently, understanding daily occurring human-centric actions, activities, and interactions by using a smartphone has drawn much interest from the research community in the context-awareness area. In this section, we give an overview of some of the work on the topic of context-awareness followed by an overview of the research work in the area of mobile clouds.

In context-awareness, the authors in [6] statee that understanding the context means inferring the situation of a person, a place, or an object. Any information from the interaction between a user and an application that can be used to characterize this situation is considered relevant. This includes the users and applications themselves. In the literature, many works tackle the problem of inferring high-level information from smartphones (see, among others [7]–[10] and references therein). A first classification of context-aware algorithms can be effectuated by considering the specific signal they work with. In particular, [7] proposes an activity recognition method designed to distinguish four different user activities by periodically classifying accelerometer signal frames using a decision tree approach. The work in [10] describes a new Location Recognition algorithm for Automatic Check-In applications (LRACI), suited to be implemented over smartphones. The algorithm uses GPS and HPS positioning information together with data received by WiFi to validate the users Check-Ins. Concerning the audio signal, [11] and [5] propose algorithms to recognize a speaker's characteristics, such as gender, identity, language, and emotional state, by using Support Vector Machines (SVMs) as classifiers.

In mobile cloud computing, there has been much research that has focused on offloading computation to reduce the toll on smartphones. The works in [2] and [12], among others, offload computation from a smartphone to a remote server in the public cloud. The main drawback of the aforementioned papers is that they rely on a working Internet connection which may not be available or feasible in many real scenarios. Mobile clouds aim to tackle this issue by relying on the devices themselves [13]. The work in [14], [15] show various examples of mobile clouds with the goal to balance the load mong devices while taking dependencies between different tasks in consideration.

III. The Transient Context-Aware Cloud

A. Transient Clouds

Transient Clouds (TCs) capitalize on the fact that smart devices can be more useful and more powerful when they work together. Due to the ubiquitous nature of these devices, anywhere with a large crowd of people will also have a large number of devices present. Importantly, these devices are not homogeneous, but each has its own unique features. TCs are designed to exploit this local non-conformity considering that the devices in the area connect to each other over an ad-hoc network while offering different capabilities for others to use. This network functions in a way similar to peer-to-peer networks, without relying on an infrastructure.

The capability that a device offers can be generic. Typically, devices can advertise their hardware features, such as having a camera or a GPS chip, but software defined features such as specific algorithms that can be executed can be capabilities as well. A device in a TC can simply request a specific capability and a device in the TC offering that capability will recruited to help out. As an example, consider a device without a GPS chip that would like to learn its location. It can simply ask the TC, and a nearby device offering a GPS chip as a capability car provide its location as an approximate location to the requesting device.

By offering these capabilities, not only can more tasks be done by more devices, but the workload can be shared across all the devices in the TC by offloading sections of code to other devices. This sharing can prolong the life of the devices in the network, rather than relying so heavily on a small number of devices that will die quickly from being overworked. For more details, we refer the reader to [4].

B. Context-Awareness as Smartphone Capability

The open, programmable and ever-growing sensor on mobile devices have enabled new sensing applications to be created across a wide variety of domains such as social networks, mobile health, gaming, entertainment, education and transportation. Devices present three different capabilities: communication capability, sensing capability and, information processing capability.

As reported in [16], they can be described by the three-trunk *"hub+sensor+processor"* paradigm. Specifically, the *hub* trunk presents both short-range (Bluetooth and WiFi employed for local information exchange) and long-range (GPRS, 3G/4G and WiFi employed as Internet access) communication capability. The *sensor* trunk is implemented through sensors embedded into smartphones such as GPS receivers, accelerometers, microphones, and radio interfaces. Finally, the *information processing* trunk is represented by the smartphones' CPUs which support flexible and powerful software. Smartphones can infer context information by acquiring data from the low level embedded sensors (microphone or accelerometer), process it to extract high-level information (who is speaking, rather than which kind of movement the user is performing) and send it to a final destination over a telecommunication network. By exploiting all these capabilities, smartphones can become powerful augmented sensors able to sense, process, and transmit high-level information, and able to characterize the situation of an entity or a group of entities and to provide information about the present status of the entities. The involved entities can be people, machine devices, objects, or locations. This new smartphone extended capability is also known as context-awareness [3].

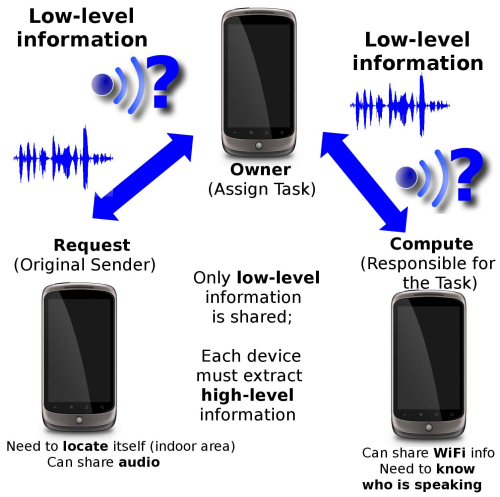

Fig. 1. Structure of the traditional TC architecture.

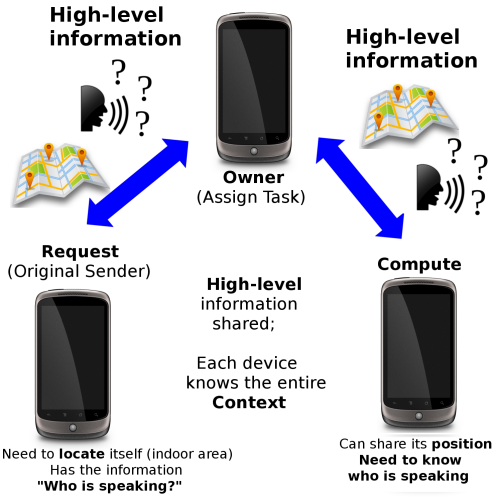

Fig. 2. Structure of the Transient Cloud-based Context-Awareness architecture.

C. Transient Context-Aware Cloud

The basic idea of TC relies on the concept of collaborative computing that allows nearby devices to form an *ad-hoc* network to provide various basic capabilities as a cloud service [4]. As reported in Fig. 1, TC permits dividing low-level tasks, leaving the network to distribute the computational load of extracting high-level information. On one hand, this could allow the users to save smartphones' battery lifetime and processing power. On the other hand, it has the limitation of only considering low-level tasks (such as sharing acquired audio or WiFi information). In this work, we merge context-aware capabilities offered by smartphones with the TC paradigm. Specifically, we have employed a solution able to recognize the speaker's main characteristics, reported in [5], named SPECTRA. It uses a Support Vector Machine (SVM) classifier jointly with some specific audio features. Together with SPECTRA we have employed the TC architecture, described in [4]. We called this new paradigm Transient Context-Aware Cloud (TCAC). Each mobile device involved in the TCAC acquires the raw data from the embedded sensors and infers high-level information which can be directly shared with other smartphones inside the TCAC platform, as reported in Fig. 2. This idea has several important and practical applications. Considering, for example, the meeting scenario in Section I, in which access to the network is forbidden (or discouraged) for privacy issues or it is simply unavailable: the TCAC platform could exploit high-level information extracted and shared by other mobile devices. If several devices are in the same room and one of them recognizes who is speaking, it can send this information through the TCAC platform, so that all the other devices could access it. Furthermore, if one smartphone infers which room the meeting is in by using an indoor positioning algorithm, all the other devices will know their position without have to compute it. This allows the devices to save resources such as energy, memory, and battery life.

IV. Performance Evaluation

A. The Test-bed Setup

To test the validity of our approach, we have employed smartphones from four different mobile device classes for the proposed experiments. Their main technical specifications are reported in Table I. The smartphones are connected together using WiFi Direct to create a TC that allows nearby devices to share high-level information. All the experiments were conducted by connecting the devices to the group owner. When the TCAC platform has extracted the speaker's details, the information is propagated throughout the cloud, so that every other smartphone can exploit it.

TABLE I. Technical specifications of the smartphones used.

Specs	Mobile Device Classes			
	Class 1	**Class 2**	**Class 3**	**Class 4**
CPU	Snapdragon 600 1.9 GHz	Snapdragon 800 2.3 GHz	Exynos 4412 1.4 GHz	Exynos 4612 1.6 GHz
Memory	32 GB 2GB RAM	32 GB 2GB RAM	16GB 1GB RAM	32 GB 1GB RAM
Android	4.4.2 (KitKat)	4.4.4 (KitKat)	4.3 (JellyBean)	4.1 (JellyBean)
Total Charge [Wh]	9.98	9.88	7.98	11.78

B. Power and Energy Metrics

In this subsection, the difference in terms of the amount of power required by the local computation and the TCAC approach is evaluated and discussed. In more detail, the power, expressed in milliwatt [mW], necessary for a single device to acquire audio samples from the smartphone's microphone and extract the features has been measured. The results are shown in Fig. 3. To measure the power consumed by a smartphone, we have written an ad-hoc Java code which links with the Android Debug Bridge (ADB). The ADB is a versatile command line tool that allows users communicating with a connected Android-powered device. It is a client-server program that includes three components: *i)* a client (which runs on your development machine), *ii)* a server (which runs as a background process on your development machine) and *iii)* a daemon (which runs as a background process on the device) [17].
Fig. 3 shows the details in terms of power consumed when all the calculations are done locally. In this plot the measurements related to certain operations are highlighted. The average value of the whole operation is about 1150[mW] (dashed red line).

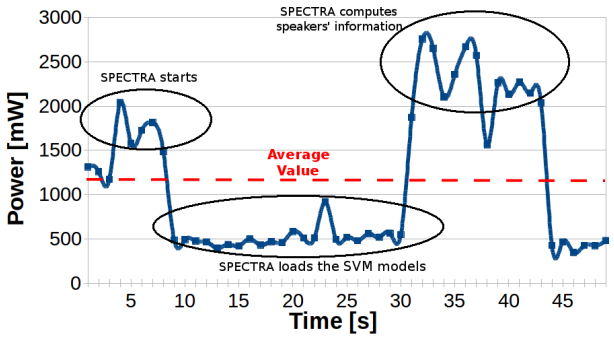

Fig. 3. Power trend of a single smartphone which computes the calculations demanded by the SPECTRA context-aware solution (see [5]) locally (continuous blue line). The average value is about 1150[mW] (dashed red line).

It is worth remembering that this value is related to the power necessary for one smartphone to infer speaker's high-level information. If more mobile devices with similar characteristics are considered, it is a reasonable assumption that each of them will employ a similar amount of power. Consequently, the average power requested by each smartphone to extract the speaker's information should be multiplied by the number of smartphones involved in the experiments in order to obtain the total power necessary so that every device has the same high-level information. This fact translates into an inefficient exploitation of the available devices' energy.

When a TCAC approach is employed, high-level information

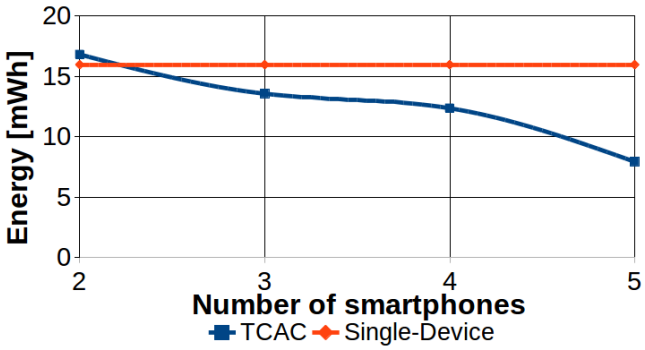

Fig. 4. Comparison of the energy consumed [mWh] as a function of the number of smartphones when *i)* all the calculation are performed locally (red line) and *ii)* all the calculations are shared within the TCAC (blue line).

(i.e., the gender of the speaker, in the case of this paper) is automatically propagated to all the devices involved in the cloud. The result reported in Fig. 4 shows the advantage of the proposed approach with respect to the case in which all the mobile devices compute their calculations locally. In particular, Fig. 4 reports the energy (in [mWh]) necessary to extract the speaker's information when *i)* all the calculation are performed locally on a single smartphone (red line) and *ii)* the calculations are shared within the TCAC (blue line), as a function of the number of smartphones involved in the experiment. From the two lines reported in the plot it is immediately clear that, when 3 or more smartphones are employed, the TCAC allows to significantly save energy, extending the devices' battery lifetime.

The next result shown, deals with the power consumed by a single smartphone when it is involved in TCAC. Reprising the TC architecture, reported in both Figs. 1 and 2, we consider the power consumed by a mobile device when it covers different roles within the TCAC. Fig. 5 shows the power consumption when the smartphone is the TCAC *Owner* device (average value 500[mW]), Fig 6 presents the power consumption for a mobile device which demands (*Request* device) the context-aware task (average value 675[mW]) and, finally, Fig. 7 reports the power consumption when the smartphone is involved to compute the final high-level information (*Compute* device, average value 101[mW]). The overall average power of three components (*Owner*, *Request* and *Compute*) of the TCAC is $\frac{500+675+101}{3}$[mW] $= 425$[mW].

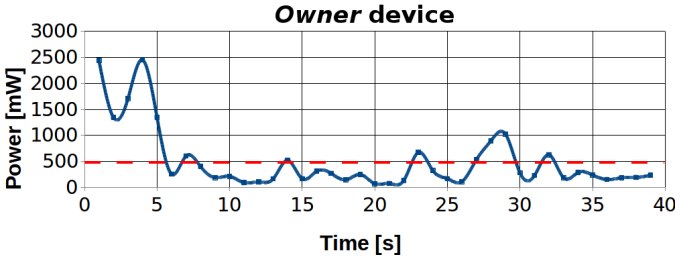

Fig. 5. Power consumption [mW] of the Owner device. The average value is around 500 [mW] (red dashed line).

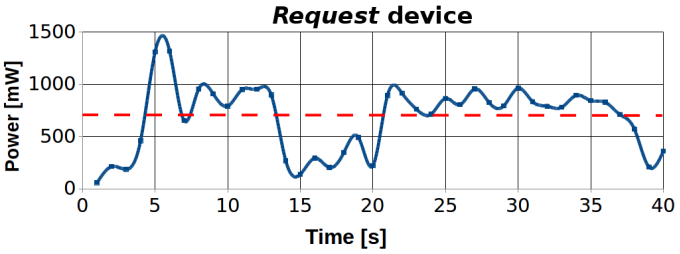

Fig. 6. Power consumption [mW] of the Request device. The average value is around 675 [mW] (red dashed line).

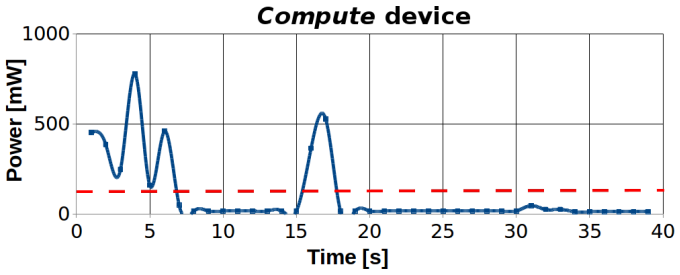

Fig. 7. Power consumption [mW] of the Compute device. The average value is around 101 [mW] (red dashed line).

The *Owner* device (see Fig. 5) presents peaks within the first seconds, due to the WiFi Direct connection management. The *Request* device and the *Compute* device have opposite power trends. From Figs. 6 and 7 it is worth noting that within the interval between 15[s] and 20[s] there is a drop in the power consumption of the *Request* device (see Fig. 6) and, in the same temporal interval, the *Compute* device exhibits a peak in its power consumption (see Fig. 7). This can be simply motivated by considering that, when the *Compute* device contributes to infer the high-level information, *Request* device can save power by offloading some computation. The

three plots reported above further confirm that when 3 or more smartphones collaborate within a TCAC, their are able to significantly save the power consumed. In fact, when the requested context-aware task is finished, each device will have available the final high-level information, with a power consumption for each device significantly lower than the 1150 [mW] necessary for a single smartphone which computes the calculations locally (see Fig. 3).

C. Accuracy of the Context-Aware Algorithm

When the TC meets a context-aware algorithm, the accuracy, in terms of the capacity of the employed solution to perform correctly, must be carefully addressed. Here, the authors have employed an algorithm able to infer some of the main speaker's characteristics (e.g., gender, identity and language) from his voice signal (see [5] for all the technical details). For sake of brevity, this work only considers the gender recognition capability, but results and conclusions can be easily extended to the other cases of identity and language. Obviously, considering that the TCAC only shares high-level information, employing the gender recognition algorithm on TCAC platform does not affect in any way the performance in terms of accuracy of the aforementioned algorithm with respect to its stand-alone usage.

Table II reports the accuracy percentage of the gender recognition algorithm embedded within the TCAC when different audio features are employed, for the Cross Validation and the Open Set scenario, respectively (see [5] for details). From Table II it is worth noting that the gender recognition algorithm shows good performances (around 98% for the Open Set scenario) for all the employed features. Furthermore, the table also highlights that the more features are employed, the better the accuracy percentage.

TABLE II. ACCURACY PERCENTAGE FOR CONTEXT-AWARE TASK (GENDER RECOGNITION) PERFORMED BY SPECTRA (SEE [5]).

	Accuracy(%)	
	Cross Validation	**Open Set**
LPC	93.89	92.8
LPC + Δ	95.37	91.86
LPC + ΔΔ	97.43	94.01
MFCC	98.31	96.78
MFCC + Δ	98.76	97.23
MFCC + ΔΔ	98.9	97.45
MFCC + LPC	96.28	93.45
MFCC + LPC + Δ	98.9	97.45
MFCC + LPC + ΔΔ	99.56	98.25

V. CONCLUSIONS

In many scenarios, it is important for the nodes in the network to compute and agree on a particular context. In some cases, it would not be worthwhile for multiple nodes to go through the same process to compute a context and in other cases, it may not be feasible. In this paper, we have presented an architecture in which context awareness can be provided over a Transient Cloud (TC). We called this new paradigm Transient Context-Aware Cloud (TCAC). Our results show that, when a few smartphones (e.g., 3 in our evaluation) are involved in the TCAC, there are significant savings in terms of energy required to perform the same calculations, so allowing to significantly extend the devices battery lifetimes. Furthermore, practical experiments show that a single device which computes the computation locally requires about 1150[mW] while, when the same smartphone is employed within the TCAC, it needs an average value around 425[mW].

REFERENCES

[1] M. Gordon, D. Jamshidi, S. Mahlke, Z. Mao, and X. Chen, "COMET: Code Offload by Migrating Execution Transparently," in *Proceedings of OSDI*, Hollywood, CA, October 2012.

[2] R. Kemp, N. Palmer, T. Kielmann, and H. Bal, "Cuckoo: A Computation Offloading Framework for Smartphones," in *Mobile Computing, Applications, and Services*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, M. Gris and G. Yang, Eds., 2012, vol. 76, pp. 59–79.

[3] J. Wu, I. Bisio, C. Gniady, E. Hossain, M. Valla, and H. Li, "Context-aware networking and communications: Part 1 [guest editorial]," *Communications Magazine, IEEE*, vol. 52, no. 6, pp. 14–15, June 2014.

[4] T. Penner, A. Johnson, B. Van Slyke, M. Guirguis, and Q. Gu, "Transient clouds: Assignment and collaborative execution of tasks on mobile devices," in *Global Communications Conference, 2014 IEEE*, Dec 2014, pp. 2801–2806.

[5] I. Bisio, F. Lavagetto, M. Marchese, A. Sciarrone, C. Frá, and M. Valla, "SPECTRA: A SPEech proCessing plaTform as smaRtphone Application," in *IEEE International Conference on Communications (ICC'15), 8-12 June 2015, London, UK*, 2015.

[6] G. D. Abowd, A. K. Dey, P. J. Brown, N. Davies, M. Smith, and P. Steggles, "Towards a better understanding of context and context-awareness," in *Proceedings of the 1st International Symposium on Handheld and Ubiquitous Computing*, ser. HUC '99. London, UK, UK: Springer-Verlag, 1999, pp. 304–307.

[7] I. Bisio, F. Lavagetto, M. Marchese, and A. Sciarrone, "Smartphone-based user activity recognition method for health remote monitoring applications." in *PECCS*, C. Benavente-Peces, F. H. Ali, and J. Filipe, Eds. SciTePress, 2012, pp. 200–205.

[8] O. Yurur, C. Liu, Z. Sheng, V. Leung, W. Moreno, and K. Leung, "Context-awareness for mobile sensing: A survey and future directions," *Communications Surveys Tutorials, IEEE*, vol. PP, no. 99, pp. 1–1, 2014.

[9] I. Bisio, A. Delfino, F. Lavagetto, M. Marchese, and A. Sciarrone, "A smartphone-centric platform for remote health monitoring of heart failure," *Wiley International Journal of Communication Systems*, 2014.

[10] I. Bisio, F. Lavagetto, M. Marchese, and A. Sciarrone, "Gps/hps-and wi-fi fingerprint-based location recognition for check-in applications over smartphones in cloud-based lbss," *Multimedia, IEEE Transactions on*, vol. 15, no. 4, pp. 858–869, June 2013.

[11] I. Bisio, A. Delfino, F. Lavagetto, M. Marchese, and A. Sciarrone, "Gender-driven emotion recognition through speech signals for ambient intelligence applications," *Emerging Topics in Computing, IEEE Transactions on*, vol. 1, no. 2, pp. 244–257, Dec 2013.

[12] E. Cuervo, A. Balasubramanian, D.-k. Cho, A. Wolman, S. Saroiu, R. Chandra, and P. Bahl, "Maui: Making smartphones last longer with code offload," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 49–62.

[13] E. Miluzzo, R. Cáceres, and Y.-F. Chen, "Vision: Mclouds - computing on clouds of mobile devices," in *Proceedings of the Third ACM Workshop on Mobile Cloud Computing and Services*, ser. MCS '12. New York, NY, USA: ACM, 2012, pp. 9–14.

[14] T. Langford, Q. Gu, A. Rivera-Longoria, and M. Guirguis, "Collaborative computing on-demand: Harnessing mobile devices in executing on-the-fly jobs," in *Mobile Ad-Hoc and Sensor Systems (MASS), 2013 IEEE 10th International Conference on*, Oct 2013, pp. 342–350.

[15] M. Guirguis, R. Ogden, Z. Song, S. Thapa, and Q. Gu, "Can you help me run these code segments on your mobile device?" in *Global Telecommunications Conference, 2011 IEEE*, Dec 2011, pp. 1–5.

[16] I. Bisio, F. Lavagetto, M. Marchese, and A. Sciarrone, "Smartphone-centric ambient assisted living platform for patients suffering from co-morbidities monitoring," *Communications Magazine, IEEE*, vol. 53, no. 1, pp. 34–41, January 2015.

[17] Android debug bridge. [Online]. Available: http://developer.android.com/tools/help/adb.html