JET: A Proof of Concept Enabling Mobile Devices as Personal Profile Providers

Javier Berrocal[1], Carlos Canal[2], Jose Garcia-Alonso[1], Niko Mäkitalo[3], Tommi Mikkonen[3], Javier Miranda[4], and Juan M. Murillo[1]

[1] University of Extremadura, Spain
{jberolm,jgaralo,juanmamu}@unex.es
[2] University of Málaga, Spain
canal@lcc.uma.es
[3] Tampere University of Technology
{niko.makitalo,tjm}@cs.tut.fi
[4] Gloin S.L.
jmiranda@gloin.es

Abstract. In recent years smartphone users have increased the number of cloud services and platforms used from them. These platforms and services are usually used, by users, to interact with others people and, by the mobile telephony firms, to create a sociological profile of the people and, thus, achieving a more adapted advertising. However, the information uploaded to these platforms is usually very similar. Uploading it to every platform entails an irrational consumption of the device resources. But, if it is not the same, the sociological profiles created could be inconsistent. The capabilities of current smartphones enable them to keep all the owner's information and to provide services for accessing it. To achieve such paradigm shift new tools and platforms are needed. This paper reports a proof of concept of a mobile application that creates and stores the sociological profiles of their users, allowing them to send messages based on those profiles. The use of this new paradigm reduces the consumption of the smartphone resources and facilitates the creation of comprehensive sociological profiles.

Keywords: Smartphones, Mobile Computing, Sociological Profiles.

1 Introduction

Penetration of smartphones in society is staggering. Taking into account only mobile phones, in a world with around 7000M inhabitants, there are more than 6800M of such devices [8]. The ever-increasing capabilities of these devices facilitate users to connect to a large number of platforms (for viewing or publishing information) and to interact with other users. This share of information is primarily done by the consumption of cloud services [3]. People being constantly connected through their mobile devices allow mobile telephony firms to obtain a significant amount of personal data from their users. This information is being used by firms to generate sociological profiles of the smartphones' owners [7].

Since multiple firms are interested in gathering these kind of information and profiles, smartphones' owners are required to upload the same data to different platforms. The replication of this information entails the irrational use of the smartphone's resources (such as bandwidth, battery, etc.) and is a risk to the privacy of such information. Thus, for example, if one has taken a special picture that wants to share with a very limited set of friends, he/she has to upload it to the common platforms used to communicate with them (such as Google+, Facebook and Twitter) and manage the different privacy options of each platform. This implies the replication of information and the waste of time and smartphone's resources. However, if all the information is not uploaded to every platform, the sociological profiles generated would be incomplete, decreasing the precision of the mobile ads shown to the smartphone's owner.

Instead of each application or service gathering different fragments of user information and storing it externally, the information can be kept in and provided by smartphones. Information about people and their real-time context can be obtained in the same way traditional services are consumed from the cloud. This new scenario implies a paradigm shift in which each smartphone provides its owners context as a service and each owner has control over his or her profile. It also enables the creation of a whole new kind of applications where this information is exploited in new ways. Additionally, not having to send all the user information to external servers, provides a significant battery saving and having all the information stored in the smartphone provides more integrated users profiles.

To achieve such paradigm shift new tools and platforms are needed. This paper reports a proof of concept of an app, called JET, providing the personal profile of the devices owners as services from smartphones. Specifically, Software as a Service is provided from these devices, where personal information about the device owner is offered. This is what we call the People as a Service [12] paradigm that aims to enable new architectures for mobile applications. In particular, the experience reported on this paper is based on a mobile communications application called Jet. This application is intended for university students and tries to improve the efficiency of current solutions by taking advantage of the information gathered by smartphones.

To detail all these aspects, the rest of the paper is organized as follows. The motivation that lead to this work is detailed in Section 2. The proof of concept reported is detailed in Section 3. Section 4 details the related works. Finally, Section 5 summarizes the lessons learned and the future steps that will be addressed in this line of work.

2 Motivation and Background

The relevance of the contextual information of smartphones owners is beyond doubt. This is shown by the interest that several of the largest software companies in the world have shown in this type of information. Such information has a high quality since these devices are mostly personal (and they are rarely used by

other people), they are always carried everywhere and they have additional sensoring capabilities (e.g. GPS and accelerometers) that are not present in other personal computing devices such as desktops or laptops.

Google is one of the most representative examples to validate this claim. Figure 1 illustrates that the main source of Google's revenue comes from online ads [4]. The dominant position Google has in this market is mostly due to the immeasurable amount of users they can display ads every day. However, just a large number of users is not all that marketing companies require nowadays. Marketing companies are increasingly demanding greater segmentation capabilities so that ads reach only those persons to whom they are addressed [9]. Google also excels at this segmentation. Mainly thanks to the huge amount of information they gather about their users.

	Full Year		2014 (unaudited)
Revenues (in millions)	2012	2013	Q1
Google Websites	$31,221	$37,422	$10,469
Y/Y Growth Rate	19%	20%	21%
Q/Q Growth Rate	NA	NA	-1%
Google Network Members' Websites	$12,465	$13,125	$3,397
Y/Y Growth Rate	20%	5%	4%
Q/Q Growth Rate	NA	NA	-4%
Total Advertising Revenues	$43,686	$50,547	$13,866
Y/Y Growth Rate	20%	16%	17%
Q/Q Growth Rate	NA	NA	-1%
Other Revenues	$2,354	$4,972	$1,554
Y/Y Growth Rate	71%	111%	48%
Q/Q Growth Rate	NA	NA	-6%

Fig. 1: Google Revenues from Advertising.

The interest some companies have in the information gathered from smartphones can be guessed in some of their initiatives. For example Facebook launched the mobile app $FacebookHome$ in April 2013. This product helps smartphones owners keep more fluid social interaction with their acquaintances. At the same time, the application served Facebook to gather large amount of information from their users for its own purposes.

Notwithstanding, gathering information from smartphones can provide benefits far beyond the strictly economic arising from marketing. Behavioural profiles of crowds can be used to improve public services' performance and planning, providing solutions towards, for example, the smart management of cities' public transportation network based on the profiles of their citizens.

However, the current strategy followed by most of the industrial approaches consisting of uploading user profiles to their own servers entails some drawbacks.

As an example, consider a user with three of the most popular apps installed in her smartphone: Foursquare, Endomondo and Facebook. Her location will be required by the three apps: the Foursqure app (if she is doing check-in), the Endomondo app (while tracking her activity) and Facebook (due to the "include geolocation" option marked while she is writing a new post). In this scenario, the same information is gathered by three different companies and sent to their correspondent servers through the available connection (suppose 4G network), with the resultant waste of bandwidth, CPU time, and energy. Additionally, users does not have a clear view of the profiles companies have from them and have little control over them. This is mainly due to the fact that smartphones play a passive role in the architecture, acting as mere sensors with capabilities of uploading information. Instead of these server-centric solutions, an alternative mobile-centric approach could make the user profile available directly from the device to every application interested on it.

Some research approaches already provide smartphones with a more relevant role that simple clients. For example, service compositions, such as *Social Devices* [14], *SociableSense* [15], *the Context Toolkit* [17], *ICrafter* [5], *Aura* [18], *Gaia* [16], *Obje/Speakeasy* [11], are good examples of this type of work.

Some of such proposals are focused on exploiting the social capabilities of smartphones. The importance of smartphones in the social computing field is undeniable. Currently, the social media services overlook the situations when people meet face-to-face and device-to-device. As a concrete example, Social Devices has introduced an app that observes when users post a new album of images to *Flickr*, and then when the user is in close proximity of her Facebook friends, the app proactively suggests a photo sharing session on the friends devices. The content and user profiles can be uploaded to external services, it is not, however, what users typically want in this kind of situations. If the content on a smartphone needs to be shared among people nearby, there is no point to upload a set of photos to a remote cloud service, wasting battery and data if, on the other hand, the content could be accessed directly from the sharing device. Furthermore, each service have their own mechanisms for managing access rights. However, if the sharing options needs to be separately configured on multiple services, managing who gets to access what, and in which context, this quickly becomes a burden for the user.

Providing services directly from smartphones raises a number of technical problems. However, some researchers already provide good solutions for them. For example Jansen's work [13] studies the technical feasibility of service provisioning from smartphones. This study shows that smartphones are able to play a more complex role than the one they play nowadays.

Following the same trend this paper presents Jet, a mobile app that allows users to communicate based on their positioning and knowledge. This app has been built by university students for university students. The app is a proof of concept proving that new mobile apps architectures are possible supported by the PeaaS paradigm. The next section details the architecture of the application and the discussion about the experience.

3 Jet

This section presents Jet, a mobile communications application that shows the viability of a new kind of mobile apps architecture based on the People as a Service (PeaaS) paradigm. We first introduce the PeaaS fundamentals and the motivations and requirements for Jet. Then we provide a technical discussion about the Jet Implementation and some discussion about the obtained results.

3.1 Fundamentals

As mentioned before Jet app is built following the principles of the PeaaS paradigm. PeaaS enables smartphones as cloud providers in which different services can be deployed for the provisioning of contextual information. This paradigm exploits mobile devices' potential for, first, storing and constructing the sociological profiles of their owners and, second, to offer the stored information by means of services deployed in them. This lets smartphones owners to keep their virtual identities under their control more easily while still allowing external systems to consult those identities. The consumers of such services can get fresh and updated information, which is also contextualized with the rest of information captured by the device.

PeaaS is proposed to overcome some of the limitations of server-centric social computing models. For that, four different principles are considered in the PeaaS paradigm. First, the users' smartphones are considered as the virtual interfaces of their owners. They connect and create virtual links with other people, or with other smartphones. Second, the smartphones' sensors and records are used to collect information about their owners. This information is used for creating the users sociological virtual profiles. Third, different services can be deployed in the smartphones provisioning the stored information, and the inferred sociological profiles, to extract and process collective sociological information, and for improving the people to people interactions. Finally, all the stored information is kept exclusively in the smartphones. Thus, users can monitor and manage who can access their information, what information can be accessed, and when it can be accessed, enhancing their privacy.

3.2 Motivations and Requirements

Nowadays, students are using communications applications such as WhatsApp, Twitter or Facebook. They use this apps to communicate about everything from their personal life to the deeper technical issues related with the subjects they are studying. This usually leads to some undesirable situations. For example, the most popular communication app used by students is *Whatsapp*. Students use this app to communicate about subjects. To use Whatsapp, students usually create different groups, varying from groups of friends with whom they often meet for fun to groups of classmates. These last groups are used to discuss about subjects topics, issues and doubts and are commonly constituted by intersecting groups of people. As there is no way to separate subjects inside Watshapp

groups, discussions start orderly but quickly become messy. Some students manifested that, during the assignments delivery seasons, one can easily get up in the morning with a thousand pendent Whatsapp messages.

Given the problem, and since they were studying subjects related with mobile apps development, some student decided to face the development of an app for communicating about subjects meeting their requirements. Thus they stated that the app should allow to communicate directly with people or with groups, that groups should be easily constituted based on subjects classmates, that discussions analysis should contribute to determine the *quality* of contributors and that the app should support sending anonymous questions that automatically should be driven to the best qualified contributors. Finally, as sometime students miss lessons and it is very important for them to contact with other student that were or are in that lessons, they manifested that it would be great if one can send messages to people regarding its past and current positioning.

Having knowledge of this initiative we decided to support them. The proposed app outlined a good scenario for PeaaS. There were two contextual data present. On the one hand, the data about users positioning and, on the other hand, data about students contributions and their qualification. Besides, the scenario would allow to show how PeaaS could coexist with traditional server-centric architectures: whilst it looks natural for us to maintain the positioning data in the mobile devices, the data about contributions could be perfectly stored in servers devoted to provide support to communications.

3.3 Jet Implementation

Jet implantation is composed of two main components: a mobile app and a backend application. Figure 2 shows the architecture of Jet. Its main components and workflows are detailed in the following subsections.

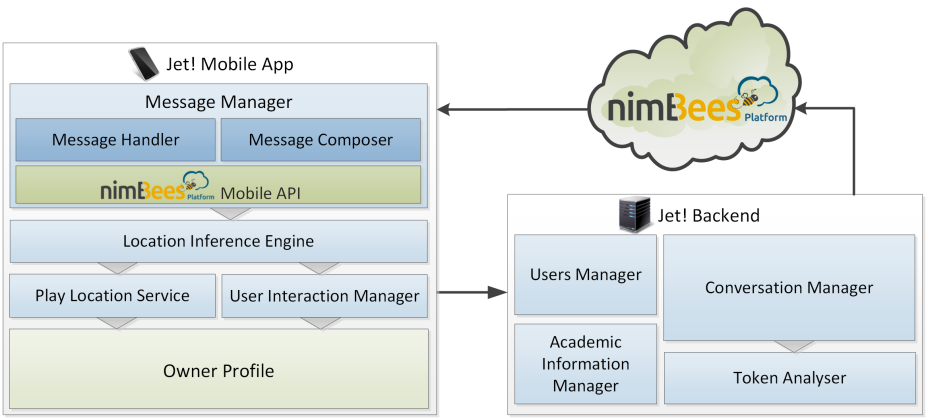


Fig. 2: Jet Mobile App and Backend Architecture.

As can be seen in the figure, Jet components take advantage of nimBees. The nimBees platform [1] is a commercial mobile push notification system based on the PeaaS paradigm.

The platform is composed of an API that, used in mobile applications, allows them to receive segmented push notifications. Once the push notification reaches the mobile device and based on the owner's profile, the API decides if that owner is an appropriate recipient for the message. Only when the owner is selected as a recipient the push notification is shown in the mobile device, otherwise all remain as nothing happened. All this transitions are transparent to the device owner.

Jet Mobile App is a mobile communications app tailored to students' needs. Figure 2 shows the main software components of the application.

When this app is installed in a smartphone it automatically starts to gather the profile of the device owner. To gather such profiles, the application uses the smartphones resources, specifically the positioning of the device. This information is gathered by the component *Play Location Service* of the architecture. For that, the component uses the Locator Manager's RequestLocationUpdates provided by Android, requesting an update when the position changes more than 100 meters. The new position is then stored in the owners profile so a timeline with date, hour and position is built. The size of the timeline is configurable and by default set up to 30 days.

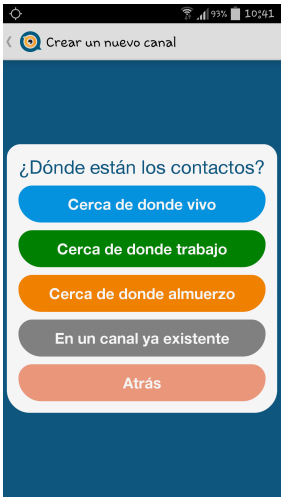
The *Location Inference Engine* component uses the timeline to infer different properties about the owner of the mobile device. In particular, it determines where the owner lives, where she/he works or study and the route she/he uses to move from home to the University. The inferred information can be visualized by its owner. Figure 3a shows the screen where it can be visualized.

The *Location Inference Engine* performs the inference routines two times per day (12:00 and 00:00) and stores the results. Each time it perform the inference, it first clean the obsolete data in the owner profile. To determine the area in which the owner lives every night the *Play Location Service* takes the position at 2:00, 3:00 and 4:00. It is considered stable if the location is the same (inside a given range) the three times. If it is stable it is used by the *Location Inference Engine* and compared with the stored residence area. If they are different, then the *Location Inference Engine* considers to change the current residence. It will do it if the new residence is maintained for at least three days.

Regarding the area were the owner works or study, the inference routine is similar to the above but, in this case, it takes the location data at 10:00, 11:00 and 12:00 (only data corresponding to working days are considered).

Finally, once the residence and working areas have been set the route followed to go from one to the other is easily inferred analysing the changes of location registered by the *Play Location Service*. Again, the route obtained each day is compared with the usual route. The usual route is changed if the new one is different and maintained for at least three days.

(a) Information inferred from the contextual data.



(b) Creation of a new channel.

Fig. 3: Snapshots of the JET app.

Jet also allows users to send messages related with positioning. In particular it allows to send messages to people that lives in an area, that uses a route to move from home to work or that were in an area in some particular date/hour. For that the capabilities of the nimBees API are used.

Particularly, when a Jet user decides to send a message, the sendNotification method of the nimBees API is called. This method will send a push notification to a highly customizable set of users. The criteria that can be used to select the users are detailed next.

- *Current location* - Filters users by their current location. Used to send messages to users that are in a specific place right now, e.g. users that are near the university cafeteria. The exact location of a device will be only consulted at the moment the notification reach it.
- *Past location* - Filters users by their location between two dates in the past. Used to send messages to users that were in a specific place at a given time, e.g users that were yesterday in the classroom at the time of a lesson I missed.
- *List of usernames* - Filters users by their usernames. Used to send messages to a set of user without forming a group, e.g. users that have failed a subject.
- *List of device identifiers* - Filters users by the identifiers of their devices. Used internally by the Jet application but never directly used by its users.
- *List of group identifiers* - Filters users by the groups they belong to. Used to send messages to one or more pre-existing groups of users, e.g. users that are subscribed to a specific subject.

To simplify the use of this API method, the *Message Composer* component of Jet raises the abstraction level, allowing its users to use more human friendly

filters. The *Message Composer* allows the users to generate new messages using the Jet GUI.

Messages can be sent directly to other users specifying their usernames, to pre-established groups (called channels in Jet) corresponding to classmates in a subject or to specific students subscribed to a course of a degree. In addition to the selection of the channel, the user can also specify location and time constraints targeting the correct recipients (for example one want to send a message to his/her Software Engineering I classmates that were yesterday in a lesson at 10:30). In that case the position is set up selecting a location in Google Maps and a radius.

The user can decide if she/he wants to send the message to everybody subscribed to the channel and meeting the time/location constraints or only to those that are experts in the topics addressed in the message. The analysis of the topics is performed using a Thesaurus in the backend side of Jet.

When a message is composed and the criteria for selecting the recipient are complete, the message is sent from the user device and handled by the Jet Backend and the nimBees Platform. After the message is processed in the backend, a silent push notification (invisible for the mobile users) is sent to the appropriate users. The nimBees mobile API in the Jet Mobile App will receive this notification and decide if the message should be shown to the user. To decide if a specific user should see the message, the *Message Manager* component uses the inferred information to take the appropriate decision.

In addition to be able to create contextualized messages, an user can create new channels in which only users with a specific profile can subscribe. Figure 3b shows the screen for the creation of channels.

The combined use of the owner's profile and the nimBees API allows Jet to use the user positioning without uploading this data to any server. Instead, the mobile device has to receive some push notifications corresponding to messages that are not really addressed to the owner. However, the cost of such receipt and evaluation of push notifications is negligible when compared with uploading the positioning data to a server. Moreover, the number of extra push notifications received is not really high due to pre-segmentation of recipients performed by the Jet backend component.

Jet Backend. It is responsible for providing support to the registration of users, routing and delivering sent messages to the correct recipients and performing the pre-segmentation of recipients. Figure 2 shows its main components.

The registration takes place the first time the user open the Jet app. During the registration process, the user sets her/his username in the app. Also, the users specify the subjects they are coursing and their groups in the subjects. This step is done by performing the appropriate selections in the academic information they are provided with. The academic information is stored and managed by the *Academic Information Manager* component in the Backend. The information corresponding to the registered users and the channels they are subscribed is stored and maintained by the *Users Manager* component.

The *Conversation Manager* component is responsible for taking requests for processing and forwarding messages. Once a sent request is received, it first ask the *Token Analizer* component to determine the topics treated in the message. It performs this actions managing a Thesaurus of the conversations. Then the pre-segmentation procedure is performed consisting in preselecting the group of appropriate recipients for the message. For that, first the users subscribed to the recipient channel are selected. Afterwards, if the message is addressed to experts, the *Conversation Manager* filters the candidates to the three users with better qualification related to the topics of the message. If the message was sent to experts and there is no answer, in a configurable period of time, the *Conversation Manager* will select the next three better qualified experts. The process continue until a reply is obtained or there are no more candidates. As in a traditional server-centric approach the conversation are stored in the backend side.

Once the definitive group of candidates is selected the messages are delivered in two phases. First a push notification is sent to the candidate recipients using the nimBees platform services. The push notification only contains the identification of the message and the time/location constraints specified for the message. As mentioned above, the Jet app in the mobile device will analyse the push notification determining if the owner is a correct recipient. Only when it is checked, the push notification is definitely displayed to the owner and the mobile device ask for the message to the Jet Backend.

As students read the messages they vote the content rewarding those replies that are more enlightening and even penalizing those most confusing. This information is used by the *Conversation Manager* to modify the qualification of the contributor for the different topics of the message. Anyway, votes and qualifications of students in the different messages and topics are kept hidden for everybody but themselves. Any student can check the qualifications they obtain.

3.4 Discussion

Jet was developed by two students. One of them concentrated on the backend side and the other one on the mobile app. We supported them by providing the nimBees platform as well as the skeleton for the mobile app with the nimBees API embedded. The implementation process took three months and after finishing, the app was tested by a group of 30 students (the 4th year of the Software Engineering Degree). The most relevant aspects observed during the evaluation of the system are discussed below.

Regarding the functionality of the application, once the testing users started using Jet they realize the benefits it provides, specially the easiness to keep the conversations focused on the starting topics, which is hard to do with other applications. This implies that some of the more social aspects of communications applications are lost in Jet, but students kept using other apps for social communications while Jet is used for professional or academic purposes.

Not uploading the positioning information to a server and instead providing it from the mobile device allows making geo-localized messaging avoiding the device battery draining. There are commercial platforms that already provide support for geo-localized push messaging. The most powerful and extended one is UrbanAirship [2]. Apps integrating the UrbanAirship API record and upload the positioning of users to a server with a configurable periodicity. The product documentation warns that setting high periodicities leads to a drastic battery draining [6]. Instead, the Jet mobile app using nimBees allows us to use the real current position of users without any appreciable effect on the battery consumption. This is due to the fact that the current positioning is only asked to those users that are candidate recipients for a message only when a message must be delivered to them. By contrast, server centric approaches periodically ask for the position to every user even when this information will never be used.

Although the students positioning information is not uploaded to any server it still remains reachable and useful. For example we are now studying the development of a new release of the app including the feature of asking for their profiles to a group of users. If the users agrees, this information will be reused by the university officials to improve the amenities offered, such as classrooms or the rooms and areas prepared for having lunch.

Jet is only a proof of concept of what the PeaaS paradigm could achieve by combining the capabilities of smartphones to store and infer sociological profiles. The main benefits come in terms of a more rational mobile devices resources consumption. However, the experiment also shows how providing users profiles as a service is also compatible with server-centric mobile apps architecture. In this case data about conversations where stored in the servers because it seemed the more natural solution: messages always pass through the servers to be routed and storing the conversation information is very useful to perform the pre-segmentation of recipients. However this information could be instead stored in mobile device. In that case there wouldn't be pre-segmentation on servers and the complete segmentation will take place in the mobile device. This would mean an extra traffic of push notification and more computation in the mobile device. In return, no user data would be on servers so their privacy would be completely managed by themselves.

4 Related work

Some research approaches already provide smartphones with a more relevant role that simple clients, improving their capabilities and increasing their importance in computational models. For example, service compositions, such as *Social Devices* [14], *SociableSense* [15], *the Context Toolkit* [17], *ICrafter* [5], *Aura* [18], *Gaia* [16] and *Obje/Speakeasy* [11] are good examples of this type of work. Although they are conceived with different proposes, the above approaches share two common features. Firstly, they are not maintaining information about the users as such, but instead maintain minimum device profiles in order to form the compositions. Secondly, the information is stored using a centralized

architecture, meaning that a single entity/server is responsible of maintaining the profiles, and other entities then query the information from there. As was discussed above, the industry has already successfully used rich user profiles to make better recommendations for the users. We believe that similar improvement can be achieved for example while suggesting interactions for the user, or while selecting devices in ad hoc service compositions.

Today, many people are using mobile messaging services like Apple's *iMessage*, *WhatsApp* and *Snapchat* for communicating with their friends. These services typically are free to use, and offer the same features as SMS messages, but may also offer some special characteristics. Snapchat, for example, is especially targeted for very instant messaging, and users can also instantly share live video simply by tabbing the screen while messaging. We believe that users choose the service based on the features it offers, and even more important, based on what services their friends use. However, as using these mobile messaging services is typically free, and using several services at the same time is possible, the threshold to change service has now become low. As an example, when Facebook bought WhatsApp many users threatened to abandon the service because with the deal Facebook got all the personal data of the users, including their phone numbers, address books, and payment information. Whether or not this really had any significant impact on the amount of users is not that relevant here, but it shows that users now have options available, and that they are also ready to change if they are not happy with their current services.

Some platforms have also experimented with context-aware messaging apps to evaluate their system. For instance, the Context Toolkit platform was used to develop a context-aware mailing list application that delivered emails for subscribers that were inside a building associated with the list [10]. Another more recent and interesting approach for proximity-based device-to-device communication is *TWIN* model [19]. In TWIN the communication is based on the device topology in wireless networks, and it offers several ways for users to contact other users, such as a radar that shows other people nearby based on the hops between wireless access points, chat, and user profiles. In a large scale user study [19], however, the users were a bit skeptical about the system and missed privacy, as in TWIN the profiles were public for everyone nearby.

5 Conclusions

Smartphones are changing the way we communicate and access different services. So far, smartphones have only played the role of service consumers. The new capabilities of these devices have allowed us to develop a new paradigm changing this role and enabling them as their owners' profiles service providers. In particular, we have reported a mobile communication application for students. This application stores and analyses the students' positioning data in their smartphones to improve the interactions among them.

The development of this system has allowed us to prove that the application of this new paradigm is entirely feasible. New architectures for mobile applica-

tions can be developed beyond of the classical server-centric approaches. Furthermore, the inferences obtained from the information in mobile devices can facilitate the creation of more comprehensive sociological profiles that can be used to provide a great social benefit. For example, PeaaS is being used to obtain the sociological profiles of patients with early Alzheimer. These profiles are later used by the doctors to make better diagnosis of these patients but, also, by their family to take care of them improving their independence and quality of life without losing security. Independently of the field in which PeaaS is applied, it shouldn't be considered only as a means of creating and accessing sociological profiles and collective information. Individuals can also take advantage of PeaaS as a model to create synergies that can have a powerful impact on society.

Currently, we are working on the integration of the *Social Devices* and *PeaaS* concepts. Social Devices enhances the proactive capabilities of smartphones to orchestrate their interactions with other devices of any kind connected to the IoT. PeaaS provides smartphones with serving capabilities that allow people to offer services from their devices. The combination of the two will allow us to develop a platform reusing the information stored in the smartphones for the coordination and interaction management of IoT devices.

Acknowledgment

This research was partially funded by the Spanish Government (TIN2012-35669 and TIN2014-53986-REDT), by the Academy of Finland (264422 and 283276) and the Nokia Foundation, and by the Government of Extremadura and the European Regional Development Fund (FEDER).

References

1. nimBees Platform, `http://nimbees.com`
2. UrbanAirship, `http://urbanairship.com`
3. Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018 (Feb 2014), `http://goo.gl/ULXROo`
4. Google Financial Tables - Investor Relations (May 2014), `http://investor.google.com/financial/tables.html`
5. Abowd, G., Brumitt, B., Shafer, S.: Ubicomp 2001: Ubiquitous Computing: International Conference Atlanta, Georgia, USA, September 30 - October 2, 2001 Proceedings. Lecture Notes in Computer Science, Springer (2001), `http://books.google.es/books?id=sRMGAQUyQEYC`
6. Airship, U.: Continuous Location Updates, `http://docs.urbanairship.com/platform/android.html#continuous-location-updates`
7. Benevenuto, F., Rodrigues, T., Cha, M., Almeida, V.: Characterizing user behavior in online social networks. In: Proceedings of the 9th ACM SIGCOMM Conference on Internet Measurement Conference. pp. 49–62. IMC '09, ACM, New York, NY, USA (2009), `http://doi.acm.org/10.1145/1644893.1644900`
8. Bureau, I.T.D.: ICT Facts and Figures (2013), `http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2013-e.pdf`

9. Canhoto, A.I., Clark, M., Fennemore, P.: Emerging segmentation practices in the age of the social customer. Journal of Strategic Marketing 21(5), 413–428 (2013)
10. Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-aware applications. Workshop on Software Engineering for wearable and pervasive computing pp. 431–441 (2000)
11. Edwards, W., Newman, M., Seciivy, J., Smith, T.: Bringing network effects to pervasive spaces. Pervasive Computing, IEEE 4(3), 15–17 (July 2005)
12. Guillen, J., Miranda, J., Berrocal, J., Garcia-Alonso, J., Murillo, J., Canal, C.: Peaas: A mobile-centric model for providing collective sociological profiles. Software, IEEE 31(2), 48–53 (Mar 2014)
13. Jansen, M.: About using mobile devices as cloud providers. In: CLOSER'12. pp. 147–152 (2012)
14. Mäkitalo, N., Pääkkö, J., Raatikainen, M., Myllärniemi, V., Aaltonen, T., Leppänen, T., Männistö, T., Mikkonen, T.: Social Devices: Collaborative Co-located Interactions in a Mobile Cloud. In: 11th International Conference on Mobile and Ubiquitous Multimedia, MUM (2012)
15. Rachuri, K.K., Mascolo, C., Musolesi, M., Rentfrow, P.J.: Sociablesense: Exploring the trade-offs of adaptive sampling and computation offloading for social sensing. In: Proceedings of MobiCom '11. pp. 73–84. ACM, New York, NY, USA (2011), http://doi.acm.org/10.1145/2030613.2030623
16. Roman, M., Ziebart, B., Campbell, R.: Dynamic application composition: customizing the behavior of an active space. In: Proceedings of PerCom '03. pp. 169–176 (March 2003)
17. Salber, D., Dey, A.K., Abowd, G.D.: The context toolkit: Aiding the development of context-enabled applications. In: Proceedings of the SIGCHI. pp. 434–441. CHI '99, ACM, New York, NY, USA (1999), http://doi.acm.org/10.1145/302979.303126
18. Sousa, J., Garlan, D.: Aura: An Architectural Framework for User Mobility in Ubiquitous Computing Environments. In: Software Architecture, The International Federation for Information Processing, vol. 97, pp. 29–43 (2002)
19. Väänänen-Vainio-Mattila, K., Saarinen, P., Wäljas, M., Hännikäinen, M., Orsila, H., Kiukkonen, N.: User experience of social ad hoc networking: Findings from a large-scale field trial of twin. In: Proceedings of the 9th International Conference on Mobile and Ubiquitous Multimedia. pp. 10:1–10:10. MUM '10, ACM, New York, NY, USA (2010), http://doi.acm.org/10.1145/1899475.1899485