Towards dynamically programmable devices using beacons

Alejandro Pérez-Vereda[1], Daniel Flores-Martín[2], Carlos Canal[1] and Juan M. Murillo[2]

[1] University of Malaga, [2] University of Extremadura, Spain
apvereda@uma.es, dfloresm@unex.es, canal@lcc.uma.es, juanmamu@unex.es

Abstract. With the grow of the Web of Things, lots of devices are being connected to the network. Many of these devices require human interaction when using them. In a desirable scenario, technology should allow to automatically adapt the behavior of these devices to the needs and expectations of their users. To this extent, in previous work we proposed the Internet of People model to automatically develop virtual profiles of people stored in their smartphones. However, in order to build a complete virtual profile with information about the user's environment and context, we need also the contribution of these surrounding devices. Our goal is to develop a framework in which users and smart devices are integrated seamlessly and in real time, allowing programmatic adaptation and update of both virtual user profiles and surrounding devices. As a proof of concept, in this paper we propose the use of beacons to dynamically download and execute in the smartphone scripts for updating the virtual profile with context information, and trigger actions both in the smartphone and the devices. This way, we take a first step to an effective Programmable World, in which everyday objects connected to the network can be programmatically adapted to their users.

Keywords: Web of Things, Internet of Things, Internet of People, People as a Service, Programmable World, Virtual user profiles, Beacons.

1 Introduction

In the last two decades, the Internet has been mainly seen as a way of connecting people, and estimations say that nowadays over half of the world's population is online [12]. However, the network is currently evolving to face a new challenge: the integration of myriads of objects in what has been called the Internet of Things (IoT) [6]. By 2020 there will be 50 to 100 billion devices connected to the Internet [17]. Indeed, hardware advances come with the creation of new chips to include in almost every device, enabling the development of *smart* things, capable of interacting with each other and with the people who use or are just near them. Most of these things rely on WWW standards for providing an interface to control and update their behavior, constituting an enormous Web of Things (WoT) [8].

Although this is not yet a problem, in a near future with huge amounts of connected devices, configuring how these smart things should work will be a serious burden for

their users. For instance, someone can configure the timetable of her home heating system to reduce its energy consumption, but if something in her schedule changes, it would be necessary to update the timetable manually. This can be done with one device, but when someone interacts with dozens of smart things everyday, this task becomes a nightmare. To avoid this situation, we require solutions which consider users' context and preferences, permitting smart things to automatically and dynamically adapt their behavior to their users.

A number of research works aim to gather and process the contextual information of users in order to build comprehensive virtual profiles [5, 10]. Among them, previous work of the authors of this paper consisted in developing a model, called People as a Service (PeaaS) [7], that takes advantage from the pervasive presence of smartphones and the wide range of sensors they include to gather information about their owners, and use it for building and maintaining a virtual profile of each individual.

However, these virtual profiles cannot be completed using only the information acquired by the smartphones themselves. They need to interact with other smart devices in their surroundings in order to achieve more precise results. For instance, a smartphone virtual assistant will suggest using the public bike service of a city to go to a certain place, only if it interacts with the nearby bicycle stations to learn whether there are bikes available.

Moreover, this interaction between user profiles and connected things cannot be reduced to the exchange of data. In their roadmap to a programmable world [19], the authors advocate for a swift from today's data-centric IoT systems to a network in which smart objects are truly programmable. Hence, we must develop general solutions in order to dynamically update both virtual profiles and the behavior of smart things in a programmatic way.

According to all this, our goal is to use people's virtual profiles for dynamically adapting the behavior of the smart devices they will use daily. In particular, we aim to develop a programming framework that offers a wide range of interaction possibilities between connected devices and people, based on their virtual profiles. In this paper we take a first step towards this goal by presenting a solution in which beacons are used to enrich the contextual information in the smartphones, even being able to dynamically program updates of the virtual profiles and to trigger different actions.

We have chosen beacons for our proof of concept as these small and simple signal emitting devices can be deployed anywhere, while other smarter devices can detect them and trigger actions from their presence. This way, beacons are a great tool to transfer context-aware behavior to any connected device.

Indeed, beacons are the devices typically used in Google's Physical Web[1]. This is an open technology whose goal is to enable quick and seamless interactions with physical objects and locations by receiving a URL linking to their web pages. However, our goal goes one step further typical Physical Web examples, as we will show.

[1] Physical Web: https://google.github.io/physical-web/

The structure of this paper is as follows. Section 2 discusses the state of the art and some related works found in the recent literature. Next, Section 3 introduces our proposal and presents the architecture of the programming framework, together with the technologies that support it. In Section 4 we develop a proof of concept to show the main features of our proposal. Finally, Section 5 presents the conclusions and briefly discusses future work.

2 Related work

The WoT offers new opportunities of orchestrating different devices to build complex systems [13]. However, as smart things are getting more presence in people's everyday lifes, this orchestration becomes a challenge. In this sense, Ambient Intelligence has emerged as a discipline for making everyday environments sensitive and responsive to people's needs [15].

If we focus on programming techniques for adapting software behavior to its context, Software Engineering approaches like Dynamic Composition [4] or Context-Oriented Programming [9] have already addressed this issue. They make it possible for developers to write a set of behaviors for a given entity, which will apply one of them depending on its external context. Although these techniques provide a certain degree of flexibility, so many variables would need to be considered during the development phase and hardwired within the code of the applications, such as user moods and preferences, those of the people around, etc. In practice, this makes it too hard to build a truly adaptable and proactive application for the WoT. The behavior and actions to take in a given situation should come up from an inference process starting off the actual context and preferences of the user.

Indeed, many recent research works agree on giving support to the WoT by means of a paradigm focused on the people [18, 20]. As mentioned before, in this paper we present a programming framework inspired by PeaaS, a mobile computing architecture that promotes empowering smartphones by giving them a key role in inferring, storing and sharing virtual profiles with personal information about their users. PeaaS is based on the Internet of People [16], a social computation model promoted by the authors of this paper that combines the IoT with people-related information to make it proactive and responsive to its users.

Many of the proposals described in the works mentioned above are related with user location. Smartphones include GPS sensors to get latitude and longitude measures outdoors, while alternative techniques [11] have been proposed to address fine-grained indoor positioning, in particular, beacons [1, 3]. In this work, we go one step forward mere positioning, and we use beacons and smartphones' Bluetooth capabilities to programmatically interact with the virtual profile stored in the smartphone.

As already mentioned, the Physical Web is based on beacons sharing the URL of a web page or application. Then, services on the smartphone, such as Google Chrome or Nearby Notifications, scan for and display these URLs after passing them through a proxy. However, we go one step further the Physical Web, as we don't use beacons just for showing web pages providing contextual information to the user, but to establish a

programmatic mechanism of interaction with the virtual profile in the smartphone, being able to modify both the information stored and the behavior of all the devices involved in the communication.

3 A programming framework for real-time context adaptation

In this section we describe the main technical features of our proposal. It relies on the use of beacons and Bluetooth Low Energy[2] (BLE) capabilities of smartphones to make them dynamically run scripts containing fragments of behavior for interacting with the virtual profile stored in them, either for updating it, for adapting the behavior of smart devices to the context of the users present in a given situation, or for both. First, we briefly discuss beacons and BLE, and then we present the details of our programming framework.

3.1 Beacons and Bluetooth Low Energy

Beacons are small devices that broadcast BLE packets allowing nearby receiving systems to determine fine-grained indoor position or to trigger some other location-related action. BLE packets can vary in size from 10 to 47 bytes and consist of several fields [14]. One of them is the Protocol Data Unit (PDU) field, which is divided in two parts: A Header and a Payload. The Header contains information about the type of the PDU out of different options included in the BLE standard, and it also indicates the length of the Payload part. The Payload represents the information actually transmitted by the beacon. In our case, it contains the URL of an online script file, a piece of code to be executed in the smartphones which detect it. For that, we use Google's open source Eddystone-URL beacon format[3] in which the Payload contains a 16 bytes field for encapsulating an URL. The format also provides some tools to shorten the URL by encoding different structures and domains like "http://www." or ".com" in one single byte.

For beacon detection, we use the Android Beacon Library[4]. This API allows mobile phones to act both as transmitters and receivers of beacon signals following the BLE standard. This way, we do not actually need physical beacons for experimenting with our proposal, but standard smartphones can be used as virtual beacons, while others act as receivers. Moreover, these virtual beacons are mobile, which allows to envision a wider range of scenarios.

3.2 Architecture of the proposal

Our goal is to develop a programming framework that: (i) makes easier the dynamic update of the user's virtual profile, adding some new appreciation to better describe

---

2 Bluetooth 4.0. Core. https://www.bluetooth.com/specifications/adopted-specifications.

3 Google's beacon platform. https://developers.google.com/beacons/overview

4 Android Beacon Library: https://altbeacon.github.io/android-beacon-library/index.html

their situation (i.e. the devices that can be detected in the nearby) and, the other way round, (ii) allows to adapt the behavior of connected devices involved in a certain situation (in a first step, just beacons), in order to trigger some actions related with the new information available from the users, such as their habits and preferences. This way, both virtual profiles and smart devices can be updated in a programmatic way, and the experience learnt will be available for the future services or devices the user interacts with.

The architecture of our proposal is shown in Figure 1. First, beacons (in fact, smartphones acting as virtual beacons) broadcast by BLE a URL pointing to a script file with the behavior we want the smartphone to execute (1). A smartphone in the nearby receives the BLE signal, and access the URL, downloading the script from a server (2). Then, the script is run by the smartphone, updating the virtual profile stored in it (3). Additionally, the instructions of the script may also indicate how to update the server's file (4), changing this way the behavior associated to the beacon.

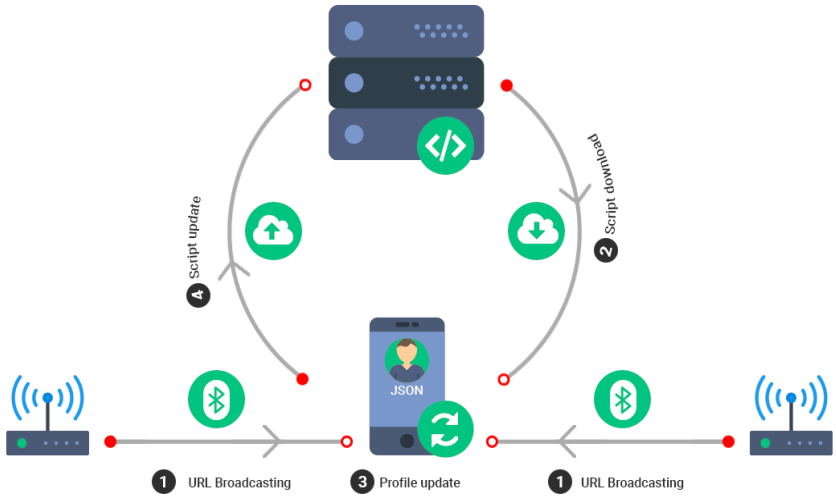

Fig. 1. Dynamic programming framework.

For the virtual profiles, we build on our previous work, and follow the PeaaS approach already mentioned in preceding sections. PeaaS endows the smartphone with the capability of storing, updating and of course sharing a virtual profile of its owner. Hence, the smartphone becomes a proxy or interface of its user with the rest of the world, in particular smart devices in the environment, allowing negotiating interactions with them in a seamless way [2]. The virtual profile can be seen as a timeline with the habits, actions and relations of the user. From the raw data collected along the day by its sensors or through user's actions, the smartphone infers the movements of its owner and the situations in which she is involved, and place them in the timeline. For instance, virtual profiles may store information about the people we meet along the day. For that,

our smartphone and those of these people must detect each other, adding the corresponding information to both profiles. This information may also include the role of each individual (e.g. a shop seller, a caregiver, etc.) and can be used to infer patterns of accompaniment.

Virtual profiles are written in JSON, which is an open and easily extensible format, and stored in the phone with Couchbase Lite[5], which offers NoSQL database storage for smartphones. Couchbase provides native APIs for iOS, Android and .NET. With these APIs we can map database documents to a native object model, work directly with JSON structures, or both. It is important to clarify that the profile of an user is only stored in her smartphone.

Finally, for running the script we use BeanShell[6], a simple Java interpreter that allows runtime upload and execution of code. A typical script states instructions for querying and/or updating a virtual profile, and probably some other actions, such as notifications to the user, etc. BeanShell scripts are just pieces of code, not complete classes, and they may make reference to external variables, which act as parameters.

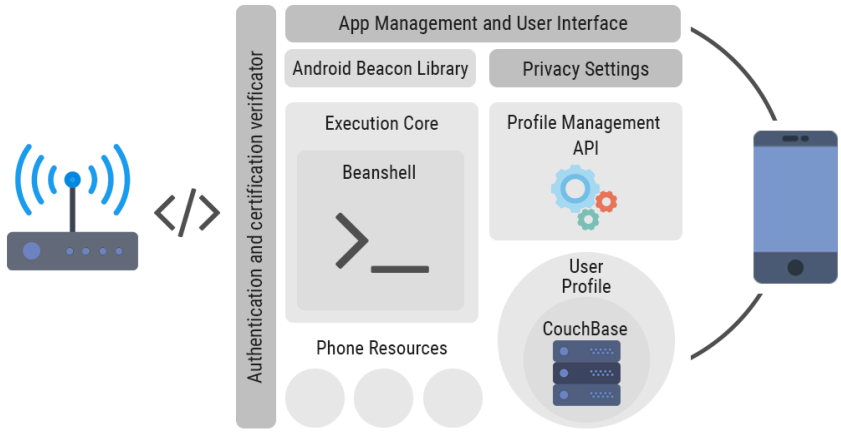

Fig. 2. Software components architecture

All these components are run in the smartphone as it can be seen in the software components architecture in the Figure 2. As it could be expected, our framework needs to pay special attention to security and privacy considerations.

With respect to security, the main concern is for sure downloading code from an unknown device and executing it in the user's smartphone. Hence, our framework includes a certification or authentication layer facing the detection of new devices. This layer can be based on certificates or a reputation system, allowing the smartphone to decide whether it has detected a trustful emitting device. Additionally, we do not allow

[5] Couchbase Lite: https://www.couchbase.com/products/mobile
[6] BeanShell: http://www.beanshell.org/

the script to execute in the smartphone risky actions as for instance a system call, nor to freely update the user profile. This is solved by providing a Profile Management API through which interacting with the virtual profile.

Privacy concerns are addressed by privacy settings which only let the smartphone to share selected aspects of the information contained in the profile. Again, these settings are taken into account by the Profile Management API, when trying to access or to modify the data contained in the profile.

4 Validating the proposal as a treasure hunting

In order to better illustrate our proposal and assess its viability, we have developed a proof of concept consisting in a treasure hunting game. Players have to find a final prize following hints hidden in treasures along their way. These treasures are represented by beacons. Each beacon has an associated script (Fig. 2) that adds to the player's virtual profile a hint to find the next treasure (lines 3-5), showing also a notification to the user (l.6). In addition, the script makes the smartphone to update the beacon file with the player's name and a timestamp (l.9-11); this way, we are able to inform future finders of the participants that have got the treasure in advance (l.12). Finally, when a player finds all the hints, a notification would be displayed telling that she has won the game (not shown in the figure), and the location of the prize. The variable `mgmt` references the Profile Management API mentioned in the previous section implementing methods for querying and updating the profile and some other utilities. The script is actually available in the shortened URL in line 1 below.

```
 1 String URL = "https://goo.gl/pzuDuu";
 2 if (mgmt.count("\"Treasure_Find_Game\" : \"treasure\"") == 1) {
 3   mgmt.append("Treasure_Find_Game", "\"treasure\": {
 4       \"value\": \"Treasure 02, Search the way\",
 5       \"hint\": \"It seems that the NORTH face is the best way\"}");
 6   mgmt.notify("It seems that the NORTH face is the best way");
 7   String name = mgmt.get("\"Treasure_Find_Game\" : \"name\"");
 8   Calendar now = Calendar.getInstance();
 9   mgmt.upload(URL, "mgmt.notify(\"player " + name
10      + " already got this hint at " + now.get(Calendar.HOUR_OF_DAY)
11      + ":" + now.get(Calendar.MINUTE) + "\")"); }
12 mgmt.notify("player Carlos already got this hint at 12:35");
```

Fig. 3. Sample script for Treasure #2 (excerpt).

Accordingly, the virtual profile of a player consists of treasures and hints that have been acquired when the player passed near one of the beacons. Fig. 3 shows the virtual profile of a user who has already got the first two treasures:

```
{... "Treasure_Find_Game": {
   "playerID" : "fn00004",
   "name" : "Alejandro",
   "treasure": {
      "value": "Treasure 01, The Beginning",
      "hint": "As in a climbing, you have to start from the ground"},
   "treasure": {
      "value": "Treasure 02, Search the way",
      "hint": "It seems that the NORTH face is the best way"}} ...}
```

Fig. 4. Virtual profile (excerpt).

Finally, for running the game we have developed a very simple Android application which scans for beacons, and downloads and runs the corresponding scripts, updating the virtual profile in the smartphone. In fact, all the treasure-hunting behavior is confined in the scripts, and the same Android application would be used for any other testing scenario we may develop.

5 Conclusions and Future Work

In previous works [7, 16], we advocated for using smartphones in order to infer virtual profiles of people with their daily activities and routines. These profiles would be one key element to a world in which technology works for the people and adapts to their needs automatically [2]. The objective is creating a seamless interface between people and their WoT environment in the Programmable World era [19]. With all this in mind, our medium-term goal is building a framework that allows programmatic update/adaptation of both user virtual profiles and surrounding connected devices. An important advantage of this approach is the development easiness intrinsic to itself as we only need one generic application for every interaction. What occurs during the interaction depends only on the scripts.

In this work, we present an initial step towards this goal. As a proof of concept, we have used beacons, as they are simple, cheap, and easy to deploy devices. However, beacons are not a key element of the proposal, and they can be replaced by any other smart device, probably presenting a more elaborate behavior. We leave this for future work.

Indeed, we use smartphones acting as virtual beacons. This also opens the possibility of using the programming framework for new collaborative scenarios in which smartphones and other smart devices can "talk", exchanging information among them, and reach common objectives.

Obviously in a framework such as the one we are proposing, authentication, trustworthiness, and security are crucial issues. We have developed a preliminary version

of the API being the only way of interacting with the smartphone and the profile persuading the script of executing non-allowed actions or modifying the profile without any control. Further development will probably imply that scripts would not be written in a general-purpose Java dialect, but in a high-level language with primitives for interacting with virtual profiles in the smartphone, and a few other functionalities, such as push notifications. Also, it is important to know in each moment with who the smartphone is interacting, as we have different trustworthiness levels depending on the person or enterprise, and of course, we must prevent from unauthorized access by malicious entities the data in the profiles and the smartphones themselves. So, we are considering a certificates-based solution for authentication in the interactions.

More detailed considerations for authentication and privacy issues have been deliberately ignored in this initial step, where smartphones simply download and run external scripts, but certainly they must be taken into account. This would be future work.

Acknowledgments. This work has been partially financed by the Spanish Government through projects TIN2015-67083-R and TIN2015-69957-R (MINECO/FEDER, UE), by the 4IE project 0045-4IE-4-P funded by the Interreg V-A España-Portugal (POCTEP) 2014-2020 program, and by the Regional Government of Extremadura (project GR15098).

References

1. Barriga-Rodríguez, A., Rodríguez-Tena, A., Garcia-Alonso, J., Berrocal, J., Flores-Rosco, R., Murillo, J.M.: Using Beacons for Creating Comprehensive Virtual Profiles. Ubiquitous Computing and Ambient Intelligence (UCAmI 2016), LNCS 10070, Springer, pp. 295–306 (2016)
2. Berrocal, J., Garcia-Alonso, J., Canal, C., Murillo, J. M.: Situational-context: a unified view of everything involved at a particular situation. ICWE 2016, LNCS 9671, Springer, 476-483 (2016)
3. Chawathe, S.S.: Beacon placement for indoor localization using Bluetooth. Intelligent Transportation Systems, (ITSC 2008), IEEE, pp. 980–985 (2008)
4. Chen, G., Li, M., Kotz, D.: Data-centric middleware for context-aware pervasive computing. Pervasive Mobile Computing, 4(2), 216–253 (2008)
5. Gronli, T.M., Ghinea, G., Younas, M.: Context-aware and automatic configuration of mobile devices in cloud-enabled ubiquitous computing. Personal and Ubiquitous Computing, 18(4), 883–894 (2014)
6. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (IoT): a vision, architectural elements, and future directions. Future Generation Computer Systems. 29(7), 1645–1660 (2013)
7. Guillen, J., Miranda, J., Berrocal, J., Garcia-Alonso, J., Murillo, J. M., Canal, C.: People as a Service: A Mobile-centric Model for Providing Collective Sociological Profiles. Software, IEEE, 31(2), 48-59 (2014)
8. Guinard, D., Trifa, V., Mattern, F., Wilde, E.: From the Internet of Things to the Web of Things: Resource-oriented architecture and best practices. Architecting the Internet of Things, Springer, 97–129 (2011)
9. Hirschfeld, R., Costanza, P., Nierstrasz, O.: Context-oriented programming. J. Object Technology, 7(3), 125–151 (2008)

10. Hong, J.Y., Suh, E.H., Kim, S.J.: Context-aware systems: a literature review and classification. Expert Systems with Applications, 36(4), 8509–8522 (2009)
11. Hossain, A.M., Soh, W.S.: A survey of calibration-free indoor positioning systems. Computer Communications, 66, 1–13 (2015)
12. International Telecommunication Union (ITU), Telecommunication Development Bureau.: ICT Facts and Figures 2005, 2010, 2014. Available at: http://www.itu.int
13. Kovatsch, M.: CoAP for the web of things: from tiny resource-constrained devices to the web browser. ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication, pp. 1495–1504. ACM (2013)
14. Mackensen, E., Lai, M., Wendt, T.: Bluetooth Low Energy (BLE) based wireless sensors. Sensors, IEEE, pp. 1–4 (2012)
15. Marzano, S.: The New Everyday: Views on Ambient Intelligence. 010 Publishers, Rotterdam (2003)
16. Miranda, J., Mäkitalo, N., Garcia-Alonso, J., Berrocal, J., Mikkonen, T., Canal, C., Murillo, J. M.: From the Internet of Things to the Internet of People. Internet Computing, IEEE, 19(2), 40-47 (2015)
17. Perera, C., Liu, C.H., Jayawardena, S., Chen, M.: Context-aware computing in the internet of things: a survey on internet of things from industrial market perspective. CoRR (2015)
18. Sheth, A.: Computing for human experience: Semantics-empowered sensors, services, and social computing on the ubiquitous web. Internet Computing, IEEE, 14(1), 88-91 (2010)
19. Taivalsaari, A., Mikkonen, T.: A roadmap to the programmable world: Software challenges in the IoT era. Software, IEEE, 34(1), 72-80 (2017)
20. Wang, F. Y., Carley, K. M., Zeng, D., Mao, W.: Social computing: From social informatics to social intelligence. Intelligent Systems, IEEE, 22(2), 79-83 (2007)