A Cloud Platform to support User-Provided Mobile Services

Vincenzo Catania, Giuseppe La Torre, Salvatore Monteleone and Daniela Panno
University of Catania - Italy
first.last@dieei.unict.it

***Abstract*—The rapid evolution of mobile computing, together with the spread of social networks is increasingly moving the role of users from information and services consumers to actual producers. Currently, since most of the critical aspects related to user generated contents have been addressed, the main issues related to service generation represent the next challenge. Dealing with services, aspects like ease of creation, discovery, security and management should always be taken into account. To cope with this kind of problems, we propose the *webinos* platform, which is based on a cloud architecture and enables user devices to share features and services among them.**

***Keywords*—webinos, cloud, user-provided mobile services**

I. INTRODUCTION

The growing popularity of Internet-enabled devices and the consolidation of social networks have increased the amount of multimedia contents generated by users. Everyday people live a second life on social networks generating original contents such as pictures, videos, comments and so on [1]. Table I contains some statistics about user content generation.

TABLE I. STATISTICS RELATED TO USER-GENERATED CONTENTS

Average amount of tweets per day	190 million
Average pictures uploaded to Flickr per minute	3000
Total amount of articles hosted by Wikipedia	17 million
Total pieces of content shared on Facebook each month	70 billion

This phenomenon has been encouraged by the spread of many kinds of Internet-enabled devices such as smartphones, tablets and entertainment devices.

Shipments of Internet-enabled devices are projected to hit 503.6 million units in 2013, up from 161 million in 2010. By 2015, however, shipments of Internet-enabled consumer devices are projected to break three-quarters of a billion units - at 780.8 million units - exceeding PC shipments of 479.1 million units [2]. Mobile devices give a new experience to users, offering them the possibility to obtain information about the surrounding environment through several built-in sensors (GPS, accelerometer, gyroscope). All these information let users create context-related contents, like geolocalized photos or tweets, which embed current user's position. A key role in this scenario is played by end-users, which are becoming the main contributors of the contents available on the web. The most likely next step in this direction will be the generation of services by non-expert users. Generating new service implies the creation of a set of API to interact with the service itself. According to the Service Oriented Architecture (SOA) paradigm, a new service could also be generated by composing one or more existing services. The result of this operation is commonly referred as "mashup". In this paper, we want to emphasize that in a not too distant future, services will be not only generated but also provided by users, primarily through mobile terminals. In particular, we refer to common users who do not have an advanced computer knowledge. A series of both software and hardware resources are necessary in order to support the user in generating and providing a service, especially if this is provided by means of a mobile device. Devices such as smartphones or tablets have peculiar characteristics due to their portability and small size. Battery life, reception problems, reduced computational and storage resources are just an example of the limitations which characterize this kind of devices. In addition, issues related to the publication of a new service, its discovery, privacy and access control raise the need of a platform to support the user in the generation and supply of services through mobile devices. In this paper, we describe *webinos*, a cloud platform for running applications and services over heterogeneous devices belonging to different domains. In the following, we will show how *webinos* can be adopted to solve typical problems of generation and supply of mobile services.

II. USER-PROVIDED MOBILE SERVICES

The aim of this section is to explain what is meant by mobile services and then outline the main issues that there are when this kind of services is provided by users through their devices. We have already said that users are increasingly involved in the generation of multimedia web content. The role of users gains even more and more importance also in the field of service generation. The emergence of Services Oriented Computing (SOC) allows end-users to develop applications by composing existing services. In this context, tools such as Yahoo Pipes [3] provide users the possibility to create own mashups composing web services. As a result, the Web is rapidly progressing towards a highly programmable platform and end-user programming has become a very popular and common trend nowadays. This enables end-users to take advantage of different Application Programming Interfaces (APIs) to create and publish their own contents and services. Major companies like Facebook, Google and eBay have already provided interfaces to their services extending their market possibilities. In this article, we focus on those services generated by users based on other applications or services provided by other mobile devices.

Mobile services are those services designed to be accessed through mobile devices. Their main aspect is the mobility for what concerns both their invocation and their supply. The difference with traditional services is remarkable: a service that

 ISBN: 978-1-61208-271-4

allows a user to view bus timetables can be provided through a web site and can be accessed in the same way on a personal computer or on a smartphone. The same service designed to be used on the move will take into account the user's context. For example the mobile service could give information for only those buses which route is close to the user's position that can be obtained through smartphone's GPS.

The potentialities of mobile services are huge. To date, there are already many context-aware applications for smartphones allowing users to benefit from mobile services. Considering the evolution of user's role from consumer to producer of content and services, is presumably that in the next few years, the average user will be able to create applications for his smartphone making a mashup of services also offered by other devices. As an example, suppose that the mobile phone owned by an elderly person provides the ability to be managed remotely. In this way, using this "device ability" a more experienced user could help the elderly to perform operations such as the remote phonebook's management.

There are several issues to consider in the creation and sharing of services across multiple devices. In particular, there would be the need of:

- A protocol to describe services and their exposed features.
- An access control mechanism to specify, through policies, the access / composition constraints of each service.
- Hosting environments (service providers) where to run services.
- Repositories where services have to be registered.
- A discovery mechanism to retrieve services (eg. by exposed features).
- A toolkit to help users to create, deploy and manage services.

In the next sections, we will give an overview of the state of the art in the field of user-generated mobile services. We will also present the *webinos* platform and how it can help to satisfy the aforementioned requirements.

III. Related Work

The scientific interest about User Generated Service (UGS) and User Generated Content (UGC) fields is growing in these last years. Zhao et al. present in [4] a comprehensive survey of current state of art in UGSs. They give the specific description of UGS by comparison with the concept of UGC, and then go through different technologies to analyze the challenges of UGS describing advantages and limitations of each approach. Jensen et al. describe in [5] some guidelines to support users creation and management of services. Tacken et al. investigate in [6] the state of the art and the requirements to let the vision of the super prosumer concept become true. They review the current technologies for an easy creation and discovery of mobile services and list the identified requirements for user generated mobile services. In [7], authors discuss the concept of mobile-services generated by the user itself. They investigate some conceptual requirements and concluded with an architecture proposal for IT service providers. Authors also provide a proof-of-concept system development performed within the European-funded project *m:Ciudad*. The European FP7 research project *m:Ciudad - a metropolis of ubiquitous services* - aims at the empowerment of users to create services on mobile terminals. The project demonstrates various scenarios in which users either act as creator of services or interact with the system to search for services or service construction components. *m:Ciudad* envisions a system for service providers, which enables a mobile user to create and consume mobile services on the fly on his mobile device. *m:Ciudad* architecture is exhaustively described in [8]. In the next section, we are going to introduce another European funded project called *webinos*. In particular, we are going to describe how *webinos* can be adopted as a platform to allow mobile service to be created and shared among users. The main advantages of *webinos* compared to other platforms will be discussed.

IV. Webinos

Webinos[9] is an Open Source Cross-Device Platform for widgets and mobile/web applications that allows developers to write applications able to run on multiple devices belonging to different domains (mobile devices, TV and automotive). In fact, the main goals of the project are applications' interoperability across devices and usability in order to create a multi-device user experience based on data synchronization and context-awareness taking into account the related security aspects.

Webinos provides a web runtime extension for browsers, which supports widget and web applications written with standard web technologies such as HTML, CSS and Javascript. *webinos* further provides a set of device-specific Javascript APIs to

- Provide access to hardware and software capabilities offered by a device such as address book, telephony manager, messaging manager, information about device status and so on.
- Access to capabilities on remote devices inter or intra Personal Zones.

The first characteristic allows developers to interact with the device, for example sending an SMS or getting geolocation and contacts information using the set of Javascript APIs. The second characteristic represents the most innovative contribution of *webinos* and allows applications running on a device to use APIs provided as services by other devices. This mechanism will be further described in the rest of this section along with a comprehensive description of the *webinos* architecture. *Webinos* introduces the concept of *Personal Zone* (PZ), defined as the set of all devices owned by a user. Each PZ has a main component called *Personal Zone Hub* (PZH), which is the point where the devices are registered and also provides data synchronization, communication among other PZs and secure access to the PZ from Internet. Multiple PZHs, one for each user, may also be linked together creating relationships among users as it happens in social networks. Figure 1 describes the overall *webinos* architecture.

Each *webinos*-enabled device placed inside a PZ has two main components called *Personal Zone Proxy* (PZP) and

 ISBN: 978-1-61208-271-4

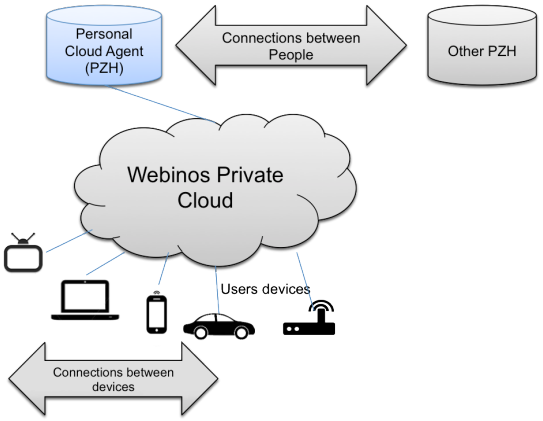

Fig. 1. An overview of the *webinos* architecture

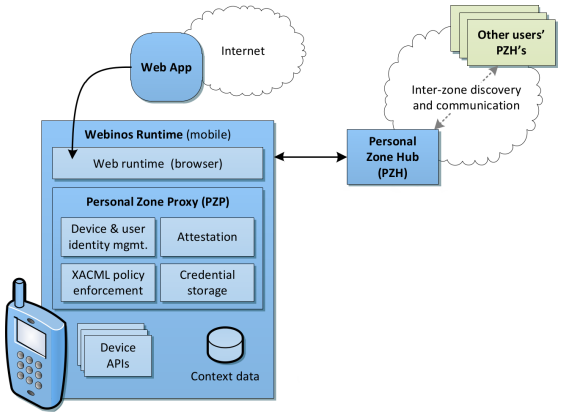

Fig. 2. Personal Zone Proxy and *webinos* runtime

webinos runtime (WRT). The WRT represents the environment where the apps are executed. *webinos* provides two kinds of WRTs: the first is a browser extension for the execution of web applications, the second is a widget runtime for the execution of locally stored applications (widgets). *Webinos* provides a WRT version for each the considered domains (mobile, PC, in-car units and home media), this means that the same application may run over all these domains without the need of a code refactoring.

The PZP connects the device to the PZH and enables the communication among devices inside the same PZ and exposes the *webinos* APIs. WRT and PZP act respectively as browser and local server, allowing each device to communicate with each other passing through the PZH (canonical way) or through a direct communication PZP-to-PZP in those situations where an Internet connectivity is not available. Also devices belonging to different PZs can communicate if their PZHs are connected. The PZH is responsible to issue identities (through PKI mechanism) and acts as messaging hub for devices and as a synchronization agent for data. User's data and services can be shared securely with other people connecting together multiple PZHs using a permission-based infrastructure. Both PZP and PZH represent the main components of *webinos* cloud architecture. Each user's content, such as an address book's contact, a calendar's event and so on, could be synchronized in every devices belonging to the user. Contents thus, are not related to a single devices but they are stored in the cloud. Although this concept is not too distant from Apple's iCloud, the most significant innovation provided by *webinos* is the possibility to share not only contents among devices but also services. In such way, devices belonging to different domains, with different OSs and produced by different manufacturers could seamlessly interoperate with each others.

Using *webinos*, users get all the benefits of a cloud platform with also the possibility to ensure privacy for their contents: *Webinos* also provides to each user the possibility to get all the benefits of a cloud platform *Webinos* provides users with all the benefits of a cloud platform offering also the possibility to ensure privacy for their contents by setting up a PZH in a private device. Figure 2 shows a detailed representation of PZP and WRT modules placed inside each *webinos* enabled device.

Other components inside PZH and PZP, called managers, are responsible for authentication, policy management, context handling, messaging, etc.

The main characteristic, which differentiates *webinos* from other apparently similar platforms such as Phonegap or Titanium or even respect mobile operating systems like Android or iOS, is the possibility to consider each API as a service provided by the device. As a consequence of this approach it is possible to create applications by invoking API on devices different from the one where the application is executed.

One of the demos presented in the *webinos* context, which mainly stands out the potentiality offered by the platform, is the *webinos Travel* application [10]. It enables user to manage his point-of-interests while a user is traveling. POIs are automatically synced between the user's devices. There is no 3rd party server integrated, where the information is stored. Syncing mechanism of the app is based on the *webinos* personal zone middleware. All data is owned by the user and resides inside zone. The application enables the interaction with the in-car navigation system. POIs can be pushed for guidance to the in-car navigation software. When the vehicle is parked, the smartphone can pick up the guidance.

V. *Webinos* AS A PLATFORM FOR USER-PROVIDED MOBILE SERVICES

Webinos introduces new scenarios for the generation and sharing of mobile services. Figure 3 shows a use-case where user has registered a personal computer and a car inside his PZ. Each of these devices has a PZP, which implements and exposes the *webinos* geolocation API. In the case of the example, a user is watching his car's position through an application running on his PC, which uses the geolocation API provided by the car. Thus, each *webinos* API implemented by a PZP can be considered as a service provided by a device. The PZP then turns each device in a server able to accept requests from other devices

Webinos provides both the mechanism for dynamic registration of new services and for discovering these services by searching the devices able to provide them. For example when a new device is added to a user's personal zone, the PZH registers all the services exposed by this new device and makes them discoverable, or not, according to the security policy set

 ISBN: 978-1-61208-271-4

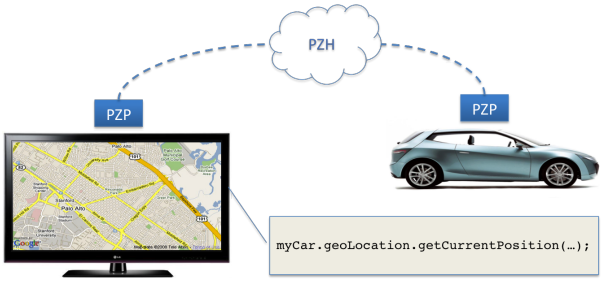

Fig. 3. An example of using "API as service"

by the user. All services provided by devices registered inside a PZ could be retrieved using the *webinos.discovery* API. We have said in the previous section that *webinos* provides the possibility to connect each other multiple PZH. Each PZH represents a user and his devices. Linking together multiple PZH means that when a user search for a service (for example the geolocation service) his PZH will query not only devices inside his PZ but also those devices belonging to linked PZs. *M:Ciudad* project considers only user generated services provided by smartphones, *webinos* instead takes into account different domains such as automotive, home-media devices and even smart objects belonging to the domain of Internet of Things. Especially in the case in which more PZHs are mutually connected, a mechanism for controlling access to services is of fundamental importance. Each PZP in fact, has an access control module based on XACML [11] specifications, which checks whether the request from an external device to a certain API may or may not take place.

Besides the possibility of calling APIs as services provided by other devices, *webinos* offers the possibility to create applications that can communicate with other applications installed on different devices. The *webinos* App2App messaging API specification defines interfaces to create, send and receive messages between applications in the *webinos* system. It provides generic messaging primitives, which can be applied in different application scenarios. The messaging is indirect, meaning that applications do not directly address each other but use a channel to route the messages to connected applications. A unique namespace (within a PZ) is used as a key to find and connect to channels. This API can be used by third-party application developers to implement custom message-based protocols by taking advantage of the features offered by the *webinos* message handling system and overlay networking model. The App2App API represents a starting point to allow the creation of new applications in the form of services, realized as a mashup of existing other services.

The possibility offered by *webinos* application to call an API exposed by another device may give rise to some problems of content management. Suppose that an application running on Alice's tablet was able to access the *webinos* Contacts API provided by Bob's smartphone to read and save locally Bob's contacts. In this case, which assumes that Bob had given access control rights to Alice, privacy concerns may arise if a third person, such as Carol, uses the Contacts API provided by Alice's tablet to read Bob's contacts.

Our future work will be exploiting the potential of *webinos* and in particular of the App2App API in order to make it possible for users to create and share *webinos* services obtained from the composition of services provided by multiple devices. In particular, we would like to

- Extend the registration and discovery mechanism to ensure that each new service created is associated with semantic information.
- Extend the current security mechanism in order to solve problems related to data handling and privacy of contents.

VI. Conclusions

In this paper, we have highlighted the metamorphosis of user's role from a simple consumer to a producer of contents and services in the Web. We described what is meant by mobile services and the problems that may arise when those services are provided through a mobile device. We also described *webinos*: a European project still that aims to define a platform for the development of user-centric applications for cross-domain targets (mobile, PC, in-car units and home media). We envision that, if properly extended, *webinos* can become the reference platform for the generation and sharing of services through users' devices.

Acknowledgments

The research described in this paper was funded by the EU FP7 *webinos* Project (FP7-ICT-2009-5 Objective 1.2).

References

[1] Statistic Brain. (2013, Feb.) Social networking statistics. [Online]. Available: http://www.statisticbrain.com/social-networking-statistics/

[2] TG Daily. (2013, Feb.) Internet-enabled devices to outpace pc shipments by 2013. [Online]. Available: http://www.tgdaily.com/hardware-features/57784-internet-enabled-devices-to-outpace-pc-shipments-by-2013

[3] Yahoo. (2013, Feb.) Pipes: Rewire the web. [Online]. Available: http://pipes.yahoo.com/pipes

[4] Z. Zhao, N. Laga, and N. Crespi, "A survey of user generated service," in *Network Infrastructure and Digital Content, 2009. IC-NIDC 2009. IEEE International Conference on*, Nov. 2009, pp. 241 –246.

[5] C. S. Jensen, C. R. Vicente, and R. Wind, "User-generated content: The case for mobile services," vol. 41, no. 12, Dec. 2008, pp. 116 –118.

[6] J. Tacken, S. Flake, F. Golatowski, S. Prüter, C. Rust, A. Chapko, and A. Emrich, "Towards a platform for user-generated mobile services," in *Advanced Information Networking and Applications Workshops (WAINA), 2010 IEEE 24th International Conference on*, Apr. 2010, pp. 532 –538.

[7] D. Werth, A. Emrich, and A. Chapko, "An architecture proposal for user-generated mobile services," in *Mobile, Ubiquitous, and Intelligent Computing (MUSIC), 2012 Third FTRA International Conference on*, Jun. 2012, pp. 142 –147.

[8] A. Emrich, A. Chapko, and D. Werth, "Context-aware recommendations on mobile services: The m:ciudad approach," in *Smart Sensing and Context*, ser. Lecture Notes in Computer Science, P. Barnaghi, K. Moessner, M. Presser, and S. Meissner, Eds. Springer Berlin Heidelberg, 2009, vol. 5741, pp. 107–120.

[9] C. Fuhrhop, J. Lyle, and S. Faily, "The webinos project," in *Proceedings of the 21st international conference companion on World Wide Web. WWW '12 Companion, New York, NY, USA*, 2012, pp. 259–262.

[10] Webinos. (2013, Feb.) Webinos travel. [Online]. Available: https://developer.webinos.org/webinos-travel

[11] OASIS. (2013, Feb.) Oasis extensible access control markup language (xacml) tc. [Online]. Available: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml

 ISBN: 978-1-61208-271-4