An Opportunistic Platform for Android-based Mobile Devices

Paolo Meroni
Computer Science Dept.
Università degli Studi di Milano - Italy
paolo.meroni@unimi.it

Elena Pagani
Computer Science Dept.
Università degli Studi di Milano - Italy
pagani@dico.unimi.it

Gian Paolo Rossi
Computer Science Dept.
Università degli Studi di Milano - Italy
rossi@dico.unimi.it

Lorenzo Valerio
Mathematics Dept.
Università degli Studi di Milano - Italy
lorenzo.valerio@unimi.it

ABSTRACT

This paper describes a novel Android-based opportunistic platform for mobile computing applications. It has the aim to incentive the growth of practical experiences that should give an answer to the following question: can Opportunistic Networks actually compete with cellular networks to support urban-wide mobile computing applications?

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication

General Terms

Experimentation

Keywords

Opportunistic networks, Android OS, mobile computing.

1. INTRODUCTION

A wide class of mobile computing applications is location-based. The ability of sieving by location the mass of information enables to feed our mobile devices with data items, such as, the list of friends in the surroundings, the closest movie theatres or the description of the nearby monument. Observed from a networking perspective, both the request and the distribution of location-based data require ubiquitous access to network services. So far, such a pervasive wireless coverage has been provided by a single technology, i.e. the cellular phone network.

There are, however, a few concerns in the use of the operators' networks as transit network for localized data distribution. The first concern is about *scalability*. Another may regard *flexibility*, e.g. many contents (opinion, ratings, etc.) with local scope are generated on the fly and could be spread in the surroundings to support the offer selection of other users. These forms of local and extemporaneous content publishing can hardly be captured by a web-based approach. A third concern is about the available *bandwidth*. Even in modern western cities, the cellular network may not always provide the adequate bandwidth to transfer large data contents. In many cases, the content is contextualized, e.g. the video of local promo, news or the photos on a smartphone; these contents can be found among the devices in the neighbourhood and their transfer could benefit of local Wi-Fi communications.

The research community in DTN considers the Opportunistic Networks (ONs) the most promising network platform for mobile computing not limited to the niche of rural applications, but able to compete with cellular networks even in an urban scenario [7]. ONs exploit the mobility of handheld devices, which come now and then into reciprocal radio range to form a mobile ad hoc network based on one-hop contacts. Although the end-to-end connectivity is intrinsically intermittent, nodes' mobility helps creating multi-hop paths leveraging one-hop contacts.

The use of ONs for our purposes is motivated by two main arguments. First, the contact-based communication is natively localized. Second, ONs favour a departure from a rigid server centric data management and adopt a peer-to-peer interaction scheme: one-hop communications are purely P2P, with synchronous peers and the contents on the peers, while multi-hop communications are still P2P, but with asynchronous peers and the content temporarily stored in the network (store-carry-and-forward paradigm).

Nevertheless, the approach shows known limitations to cope with. Although consensus exists on the fact that ONs can be used to locally diffuse queries of the data and their responses [5], much work must be done to reduce the latency in between through proper search, cache and pre-fetch mechanisms that enable to speed up the query processing.

In this application scenario, a central role is played by the handheld devices that are demanded to provide the middle-

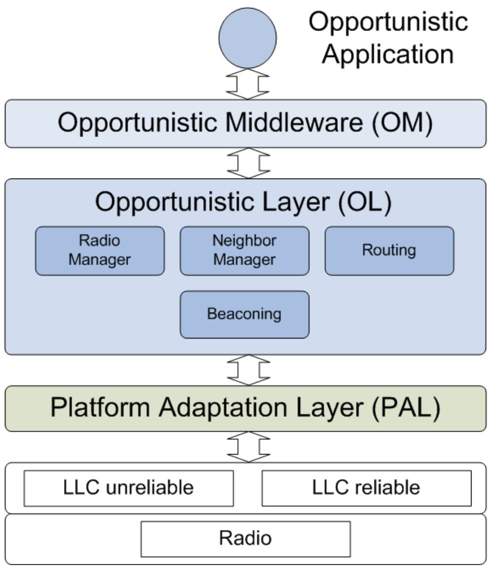

Figure 1: The Android-based Opportunistic Architecture.

ware features to access the data items and the networking capabilities to operate opportunistically.
This paper describes the implementation of a novel Android-based opportunistic platform for mobile computing. It exploits the recently unveiled Google Android OS distribution [3] and has the aim to incentive the growth of practical experiences that are needed to give an answer to the following question: can ONs actually compete with cellular networks to support urban-wide mobile computing applications?

2. OPPORTUNISTIC ARCHITECTURE

The Android-based opportunistic Architecture is composed of the following components: the **Opportunistic Middleware (OM)**, the **Opportunistic Layer (OL)** and the **Platform Adaptation Layer (PAL)**, as shown in Fig.1. These layers operate on top of the Logical Link Control and the Radio. The current implementation adopts Wi-Fi as the unique radio technology.
This paper mainly focuses on OL and PAL Layers, that are both currently implemented, while OM is still in progress. In practice, all the applications used in our opportunistic test bed, have the OM services embedded into the application itself. Purpose of OM is to provide direct support to the communication needs of the user applications through services such as, (i) the naming of resources, data items and peers, (ii) the management of queries/responses and, (iii), of P2P sessions. A few proposals exist that cover the same overlay functionalities [1, 5]. To enable the porting of different OM platforms onto the Android framework, an API should provide the clean interface between a generic OM and the underlying OL functionalitites. This is the approach we followed. The available API allows the access to the services of three main OL components:

- *Radio Manager (RM)*, which allows to create, delete and maintain a single-hop ad hoc wireless network, and hides all dependencies related to the radio channel;
- *Neighbours Manager (NM)*, which provides the one-hop contacts membership;
- *Routing*, which, at present, implements on the base of a store-carry-and-forward policy the epidemic routing strategy defined in [8].

To quickly adapt to the changing neighbourhood, the NM uses the level-3 *Beaconing Service (BS)*, that periodically broadcasts HELLO packets to all one-hop encounter nodes and listens to their beacons. Each HELLO packet contains the node ID and has some extra space for future use (e.g. to collect further information needed by more specific routing policies). The **Platform Adaptation Layer (PAL)** provides an abstraction of all machine and communication dependencies. The approach is clearly motivated by the need of improving software maintenance and portability. The PAL interfaces with both reliable and unreliable LLC services. The former is mainly used to support the one-hop transfer of large data items such as, photos and files. The latter is used, for instance, by the beaconing protocol.

3. SIMPLE USE CASES

We are validating the described opportunistic platform by means of a simple application that shows on the screen of a mobile device the avatars of persons in the neighbourhood who match some specified attributes. Each mobile user is described by a user profile that might, for instance, include the resources s/he is available to share. In practice, this application simply translates into a P2P and a location-aware (e.g. workplaces) paradigm, what well known server-centric social network applications offer to their mobile users in an urban scenario. The location scope is specified by the number of hops the data are allowed to travel before being dropped. When the minimum threshold of one hop is specified, only the nodes in direct radio contact are considered. When changes in the neighbourhood are notified by the Neighbour Manager to higher layers, the profile is exchanged between entities of the OM. The OM processes the received profiles, and diffuses them outwards via the Routing service as far as specified by the hop count. Only profiles matching the locally-defined attributes are passed to the application for avatars displaying (Fig.2).
We can, for instance, consider two simple use cases. The first allows to share contents with the encounters while walking in a workplace or meeting someone (one-hop scenario). The second allows to know when all members of your research group have entered the Department to schedule an extemporaneous meeting (multi-hops scenario).
Let us now focus on the first scenario to shortly describe the sequence of events that are generated at different layers when the application is running. Once activated, the application issues an explicit request to join the opportunistic network. This task is performed by the *Radio Manager* that will return the connection handler to the application and activates the *Neighbour Manager*. The NM maintains its neighbour list by exploiting the services of the *Beaconing Service* and waiting for its notifications. In the current implementation, BS sends one-hop HELLO packets composed of the device network address only, receives the HELLO packets of other encounters and notifies NM. NM maintains the list of all encounters and assigns a time to live to each entry. Should this time expire, then the entry will be removed. The time to live is refreshed whenever a HELLO packet is received by the given encounter. At present, BS

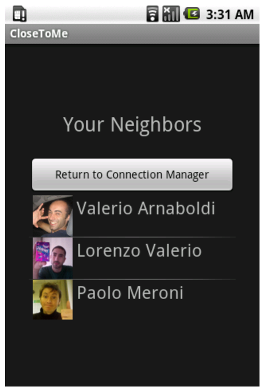

Figure 2: Application Screenshot

sends HELLO packets with a cycle of 1 sec. and the time to live is set to 10 consecutive BS cycles. NM notifies upwards all the neighbour list changes. The application establishes a P2P session with any new encounter to send its profile. The current profile simply contains the photo and the name of the encounter. Both information is displayed on the mobile device. The encounter's profile (and the information displayed) are removed when the encounter moves out of range. The opportunistic architecture just described has been implemented on Android OS version 1.6. The devices involved are HTC G1, equipped with a Qualcomm®528 Mhz processor, 192MB of RAM and a 802.11b/g Wi-Fi device. The implementation required to cope with several critical issues. Two of them deserve a short mention: (1) although the HTC-G1 Wi-Fi driver allows it, the current Android development policies do not enable the creation of networks in ad hoc mode. An unofficial patch to the Android OS [2] helps to avoid the problem. (2) The connection state is monitored by the wifiStateTracker private object, whose current observation can only be accessed by instantiating a wifiInfo public object. The obtained value, however, is a static view of wifiStateTracker. This forces the programmer to instantiate a new wifiInfo object every time he needs to know the current connection state, with the consequent waste of resources [4].

4. PRELIMINARY EVALUATION

Both the user experience and the application performances may be influenced by the time required to join the opportunistic network, i.e., to establish one-hop contacts. If the join time is too long (e.g. 10-12" as in Bluetooth) then many contacts lasting few seconds will be missed. In this case, performances may be negatively affected as short contacts can fasten both data diffusion and forwarding [6]. To this purpose, we measured the join latency[1] for different radio ranges. This experiment has been performed in a rural area to avoid any kind of radio interferences. As shown in Fig.3(a), the observed average delay of 570 *ms* is very small and almost independent of the radio range, while Fig. 3(b) shows that the distance among radio devices negatively influences the ratio of join failures.

At present, the implementation has not considered the issue of energy conservation and the beaconing process keeps the radio always on. Despite specific measurements have not been performed so far, we observed that one-day experiments, with handheld devices frequently coming in and out the radio range, exhausted nearly the 75% of the battery charge.

[1]With join latency we mean the elapsed time between a join request and the join notification to the application

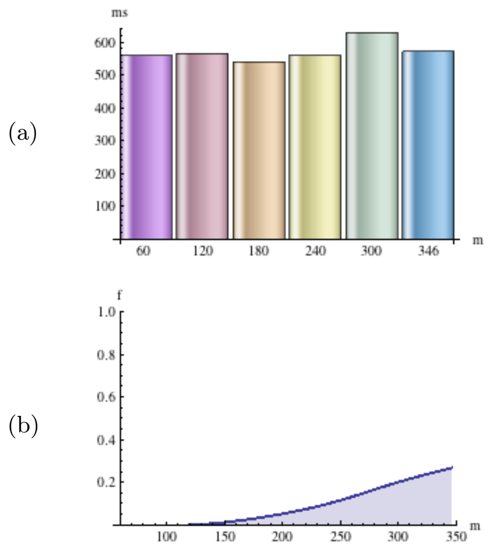

Figure 3: (a) Mean join latency vs. radio range; (b), Ratio of join failures vs. radio range.

5. CONCLUDING REMARKS

This Android-based opportunistic platform is sustained by the belief that human mobility, and the resulting contacts among people, represent a viable connectivity infrastructure for a wide class of mobile computing applications. However, such a belief must be confirmed through practical experimentations. This is what we hope this opportunistic platform will enable.

6. REFERENCES

[1] Ace toolkit. http://acetoolkit.sourceforge.net/.
[2] Adhoc support for wi-fi. https://review.source.android.com/9714.
[3] Android. http://source.android.com.
[4] Google code, android, issue 3641. http://code.google.com/p/android/issues/detail?id=3641.
[5] F. De Pellegrini, I. Carreras, D. Miorandi, I. Chlamtac, and C. Moiso. R-p2p: a data centric dtn middleware with interconnected throwboxes. In *Autonomics '08*, pages 1–10. ICST.
[6] S. Gaito, E. Pagani, and G. P. Rossi. Opportunistic forwarding in workplaces. In *Proc. 2nd ACM SIGCOMM Workshop on Online Social Networks*, 2009.
[7] A. Lindgren and P. Hui. The quest for a killer app for opportunistic and delay tolerant networks. In *Proc. 4th ACM CHANTS Workshop*, pages 59–66, 2009.
[8] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, Duke University, 2000.