TinyBox: Social, Local, Mobile Content Sharing

Carlo Caprini, Mattia Zeni, Nicola Conci
University of Trento
via Sommarive 5, 38123 – Trento, Italy
{*carlo.caprini,mattia.zeni.1*}*@studenti.unitn.it*
conci@disi.unitn.it

Iacopo Carreras, Daniele Miorandi
CREATE-NET
via alla Cascata 56/D, 38123 – Trento, Italy
{*icarreras,dmiorandi*}*@create-net.org*

***Abstract*—TinyBox is a mobile application for sharing contents within local social groups. TinyBox makes use of opportunistic communications, being able to exploit various connectivity options and to work in the absence of Internet connection. TinyBox allows users to easily create and manage shared repositories for multimedia contents, making use a rule engine for defining sharing attributes and rights. It features a peer-to-peer synchronization manager for ensuring up-to-date contents are available to all participants in the group. TinyBox has been developed for both Android (smartphone version) and Mac OS 10.8 (laptop version). In the demo we will show live the functionality of the application, including its ability to work in the absence of Internet connectivity and to effectively exploit opportunistic communications, in particular when synchronizing multimedia files.**

***Keywords*-swopportunistic communications, file sharing, synchronization**

I. Introduction

A number of services exist today for sharing files and contents among groups of selected users. Content sharing platforms target mainly two types of usage.
On the one hand, there are solutions for creating shared repositories, to be used by groups working on a common project for exchanging data, knowledge and information. Examples include Dropbox[1], SkyDrive[2] and GoogleDrive[3]. All such solutions are typically cloud-based, where the master copy of a shared file is kept on a remote server and all clients access it for obtaining the latest version of the file. Synchronization is run in the background, with users uploading a revised version of a file to the cloud and a central controller notifying the update to the clients, which then download a local copy of the new master version.
The second class of content sharing solutions is focused on social groups. Such feature is commonly embedded in social networking platforms, such as Google+[4] with its notion of 'circles'. A number of mobile applications also exist, which focus on 'local' exchange of contents, where geographical proximity constitutes the trigger for content exchange and synchronization. Examples include Bump[5] and Chirp[6].

In this demo paper we introduce the TinyBox mobile application for the sharing of contents and files within local social groups. TinyBox has been designed to opportunistically exploit communication opportunities, being able to exploit various wireless technologies and to work even in the absence of Internet connectivity [1], [2].

TinyBox allows users to easily create and manage groups for the social sharing of sets of contents (called "virtual rooms" in the laptop version and "experiences" in the smartphone one), making use a rule engine for defining sharing attributes and rights. It features a peer-to-peer synchronization manager for ensuring up-to-date contents are available to all participants in the group while minimizing the amount of data to be communicated [3], [4].

TinyBox has been developed for both Android (smartphone version) and Mac OS 10.8 (laptop version). In the demo we will show live the functionality of the application, including its ability to work in an Internet-less setting as well as to effectively exploit transmission opportunities, in particular when synchronizing large multimedia contents.

II. Research Contributions

In this Section we briefly discuss the key innovative aspects of the TinyBox application.

- **Opportunistic Communications:** TinyBox has been designed in order to natively supporting opportunistic communications, taking advantage of network resources as and when they become available [2], [5]. In the presence of a known WiFi network, the application makes use of local UDP broadcast messages to announce its presence as well as to discover nearby users. In the absence of known networks, the devices running the TinyBox application launch a back-off timer, after which they set up a WiFi ad hoc network. The SSID of the network established is defined within the application code, so that users running TinyBox join it as soon as it is made available. The process is autonomic and totally transparent to the end users. A block diagram

[1]https://www.dropbox.com
[2]https://skydrive.live.com/
[3]https://drive.google.com
[4]https://plus.google.com
[5]https://bu.mp/
[6]http://chirp.io/

978-1-4244-9529-0/13/$31.00 ©2013 IEEE

of the mechanism developed in sketched in Fig. 1. The net result is the ability to share contents within local groups even in the absence of Internet connectivity.

- **Rule-based Sharing Groups Management:** In TinyBox shared experiences and virtual rooms are defined by means of a set of rules. Rules can apply to both contents and users. Users matching a given set of rules are granted permission to access or edit shared files and folders. Rules can be dynamically changed at runtime by the group creator. This approach provides a level of flexibility not supported by most file sharing applications, where users have to select manually both the content and the users to share this content with. An example of a TinyBox group description would read as follow: "access granted until tomorrow to the friends currently in my proximity to all pictures taken in the last 20' from my phone camera".
- **Synchronization of large files:** given its strong focus on the support of opportunistic communications paradigm, synchronization has been designed to be inherently tolerant to disruptions and able to work effectively in the presence of transfer interruptions. In particular, attention has been paid to ensuring that large files (typically: multimedia ones such as videos) can be effectively transmitted. The chunk-based `rsync` [4], [6] mechanism is used in order to minimize the data to be transmitted for updating a shared file. File transfers that started but were not completed due to a connectivity problem are resumed from the point where they got stopped.

III. Implementation Aspects

TinyBox has been prototyped in two versions. One targets laptop users, and has been specifically developed in Objective C for the Mac OS 10.8 operating system. Target use cases for the laptop version include support to ephemeral localized working groups, which need to share files on-the-fly in the absence of Internet connectivity. The second one, which will be demonstrated at the conference, targets mobile users and has been developed in Java for Android devices. Target use cases include sharing of pictures and videos with friends in proximity.

A few sample screen-shots of the laptop version (Fig. 2) and the smartphone version (Fig. 3) are here reported.

IV. Demo Setup & Requirements

In the demo we will showcase the main functionality of the TinyBox application. In particular: its ability to work in the absence of Internet connectivity, the experience/virtual room creation and sharing features and the capability of optimizing the transmission of large files in the presence of connectivity disruptions. The demo will be focused around the smartphone version of TinyBox.

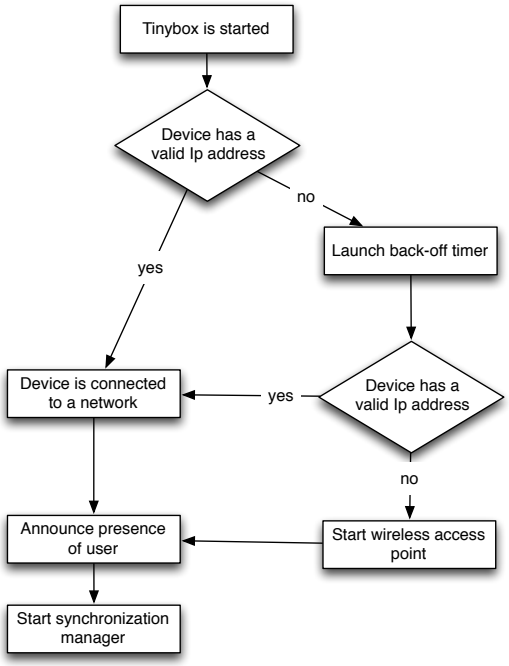


Figure 1: TinyBox: block diagram of the mechanism supporting opportunistic communications.

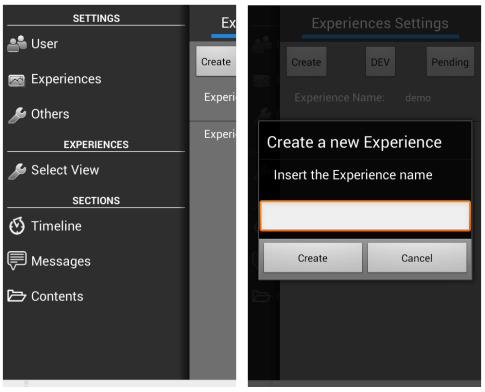
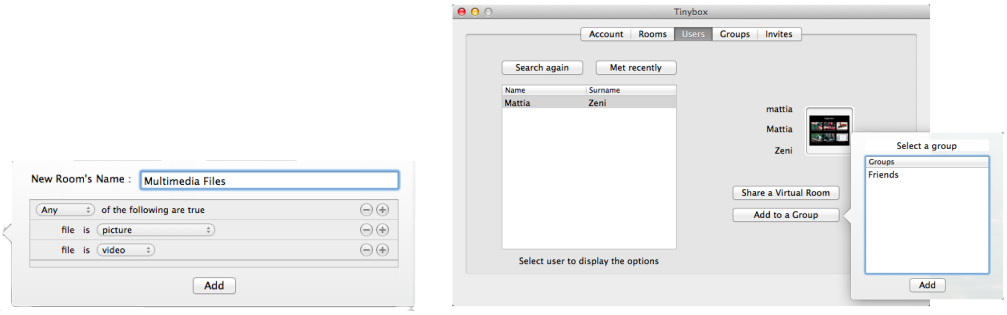


Figure 3: Screen-shots of the TinyBox application (Android version).

A. Practical setup

The demonstration scenario, represented in Fig. 4, consists of three users, Alice, Bob and Charlie. All of them have a smartphone, and are in proximity of each other. Four use cases will be demonstrated.

In the first use case, Alice wants to share with Bob a picture she just shot. So Alice creates a group on the fly, providing Bob with the right to access all pictures taken within the last few minutes. Given a wireless access point, Alice's TinyBox application sends a UDP broadcast message notifying the new group. The TinyBox application running



(a) Rule-based creation of a virtual room.

(b) Managing users, groups and virtual rooms.

Figure 2: Screen-shots of the TinyBox application (laptop version).

on Bob's phone receives the message, authenticates with Alice and gets the file.

In the second use case, Charlie wants to share a picture with nearby friends. However, his device does not recognize any of the WiFi networks available as a known one. The TinyBox application hence automatically sets up a WiFi ad hoc network, through which the file gets transferred to Bob.

In the third use case, Alice wants to share a video with Charlie. During the transfer, Charlie moves out of the WiFi network coverage, interrupting therefore the transfer. When he gets back into reach of the WiFi network, the transfer gets resumed from the point where it got interrupted, effectively concluding the file sharing.

In the fourth use case, Bob applies a nice 'Rise' filter to a picture he got from Alice. Happy with the result, he saves the new version of the picture. Thanks to the TinyBox synchronization engine, Alice immediately gets the updated version of the picture.

The demo setup will comprise one WiFi access point and three Android-based smartphones, which will be used to showcase TinyBox capabilities to the conference's attendees. A laptop will also be present and will run a short presentation describing the TinyBox operations and providing insight into its design and implementation. A poster with a high-level description of the application will also be shown.

B. Technical requirements

This demonstration will require a booth with space for hanging an A0-size poster. A desk is needed, with space for a laptop and a WiFi router (both will be brought by the authors). Four power outlets are required. No Internet connectivity is needed.

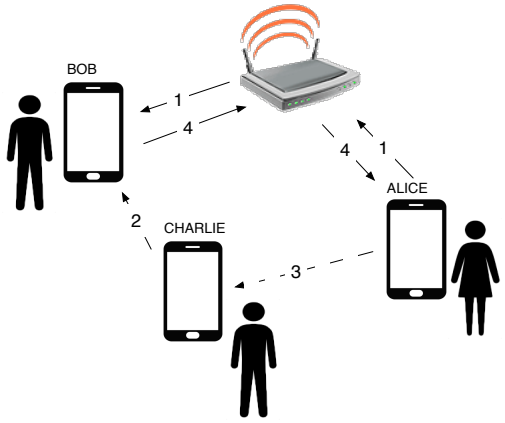


Figure 4: The live sharing scenarios considered for the demo. Number on arrows identify the four use cases that will be presented.

References

[1] S. Keshav, "Design principles for robust opportunistic communication," in *Proceedings of the 4th ACM Workshop on Networked Systems for Developing Regions*, ser. NSDR '10. New York, NY, USA: ACM, 2010, pp. 6:1–6:6.

[2] L. Pelusi, A. Passarella, and M. Conti, "Opportunistic networking: data forwarding in disconnected mobile ad hoc networks," *IEEE Communications Magazine*, vol. 44, no. 11, pp. 134–141, November 2006.

[3] T. Suel, P. Noel, and D. Trendafilov, "Improved file synchronization techniques for maintaining large replicated collections over slow networks," in *Proceedings of the 20th International Conference on Data Engineering*, ser. ICDE '04. Washington, DC, USA: IEEE Computer Society, 2004, pp. 153–164.

[4] A. Tridgell, "Efficient algorithms for sorting and synchronization," Ph.D. dissertation, The Australian National University, 1999.

[5] K. S. Phanse and J. Nykvist, "Opportunistic wireless access networks," in *Proceedings of the 1st international conference on Access networks*, ser. AcessNets '06. New York, NY, USA: ACM, 2006.

[6] A. Tridgell and P. Mackerras, "The rsync algorithm," Australian National University, Department of Computer Science, Technical Report TR-CS-96-05, Jun. 1996, (see also: http://rsync.samba.org).