The Design and Implementation of a Distributed Photo Sharing Android Application Over Ad-Hoc Wireless

by

HaoQi Li

Submitted to the Department of Electrical Engineering and Computer Science
in Partial Fulfillment of the Requirements for the Degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

Massachusetts Institute of Technology

June 2012

Author .....
Department of Electrical Engineering and Computer Science
May 21, 2012

Certified by .....
Li-Shiuan Peh
Associate Professor of Electrical Engineering and Computer Science
Thesis Supervisor May 21, 2012

Accepted by .....
Prof. Dennis M. Freeman
Chairman, Masters of Engineering Thesis Committee

The Design and Implementation of a Distributed Photo Sharing Android Application Over Ad-Hoc Wireless

by

HaoQi Li

Submitted to the Department of Electrical Engineering and Computer Science
on May 21, 2012, in partial fulfillment of the
requirements for the Degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

TODO/Ask: I'm using "WiFi" to mean different things here, what should they be?
TODO/take out word count. Word count: 136

We present a distributed photo-sharing Android application, CameraDP, that relies on ad-hoc wireless over WiFi. The app utilizes the novel DIstributed Programming Layer Over Mobile Agents (DIPLOMA) programming layer to provide a consistent shared memory over a large distributed system of android phones. The success rate and latency of photo saves and photo gets on CameraDP were compared to the numbers generated from CameraCL, a WiFi-only version of the same user interface as CameraDP. Under near-ideal WiFi conditions with only a 1.4% sacrifice in success rate, a 10-phone CameraDP system yielded a 2.6x improvement over a 10 CameraCL phones running on 4G, and the CameraDP system yielded a 16x improvement over CameraCL running on 3G. The ideas and methods of this research could be beneficial in the future if WiFi becomes more robust and smart phone WiFi ranges increase.

Thesis Supervisor: Li-Shiuan Peh
Title: Associate Professor of Electrical Engineering and Computer Science

Acknowledgments

I would like to thank my thesis advisor Li-Shiuan Peh for her clear explanations and valuable comments, my labmates Anirudh Sivaraman and Jason Gao for their debugging assistance, and all those who volunteered for the experiments, thank you for your patience.

Contents

Chapter 1

Introduction and Motivation

Smart phones heavily rely on the Cloud to carry out extensive computations or get access to abundant storage. Frequent communication with the Cloud, especially in an area dense with smartphones, can cause problems such as decreased bandwidth, decreased response time, and reduced battery life, among which the increased latency time causes an immediate frustration to the users.

A theoretical solution to this problem is to set the phones in a distributed shared-memory network. Given a good enough WiFi condition, requests to nearby phones should be faster than requests to the cloud, increasing user experimence on the phone. This is an area of active research and we utilize a recently built consistent shared-memory system, DIPLOMA, to test the feasibility of a popular photo-sharing app, Paranomio, on a distributed memory system relying mostly on the ad-hoc WiFi.

We created a stripped-down version of Paranamio that only have two functions: *take* new photos and *get* other's photos. In order to quantify the advantage of ad-hoc WiFi, we built two functionally identical apps: CameraDP and CameraCL. CameraDP uses DIPLOMA in the background while CameraCL is the control case where each request is independently sent to the cloud. one using DIPLOMA (CameraDP) and the other using 3G or 4G.

Chapter 2

Background on DIPLOMA

Chapter 3

User Interface and Functionality of both Camera Apps

The goal of the experiment is for users to share photos among themselves using their smart phones. The smart phones can take new photos and request the latest photos taken by other phones. But unlike traditional photo sharing apps where each phone functions individually, our experiment assigns the phones into different regions based on their GPS locations and a regions phones collectively save their photos. This implies: a) a new photo is saved on its phones region, not the phone itself and b) a phone can only request the newest photo of a region, not of another individual phone.

In our experiment, six square consecutive regions (0 to 5) about 30 meters wide were created along a stretch of busy road. Ideally, there is no limit to the number of regions as long as a phone can only belong to one region at any time. The size of the region should be as big as possible, but still small enough that phones from any points in the region are within WIFI ranges from each other. The users walked around and pressed buttons on the apps to take new photos or request photos from different regions. With two apps on two different phones, the users synchronized the button presses so that sequences of events for the two app types are similar.

The two different apps are: DiplomaCamera and CloudCamera. Even though their UI are identical, DiplomaCamera handles the pictures using DIPLOMA while CloudCamera is the control of the experiment, handling all requests with the cloud.

Chapter 4

CameraDP Android Application

The DIPLOMA design introduces a leader in each of the regions. The leaders take care of all the requests coming from all phones (clients) in the region. When a phone takes a new photo, it broadcasts the photo data to its leader, where the photo is saved. When a phone requests a photo from Region X, this request is broadcasted to its leader, which in turn uses DIPLOMA to relay the request and receive Region Xs photo data from Xs leader.

The code is divided into three big components:

1. StatusActivity.java for UI and client processing 2. UserApp.java for leader and remote leader functions 3. The DIPLOMA java files: Mux.java, VCoreDaemon.java, DSMLayer.java are unchanged.

StatusActivity.java contains listeners for the button presses that send requests to its region leader and a handler that processes replies from the region leader. Each phone has a unique id based on its IP address that can help a regions leader distinguish the non-leader phones in its region.

Pressing the button that takes a picture triggers that buttons listener to retrieve the photo information from the Camera SurfaceView. The photo data is then put into a packet along with the phones ID, the phones region number, and type of request (UploadPhoto). This packet is serialized inside StatusActivity.java into a UDP broadcast that reaches the leader of the region.

Similarly, pressing a button that requests the newest photo from region X triggers

the request buttons listener to get information on the target region number that the user is requesting. A UDP packet consisting of the phones ID, the phones region number, the target region number, and the type of request (DownloadPhoto). Again, StatusActivity.java broadcasts this packet to the leader of the region.

Let Original Leader (OL) be the leader of the phone that made the request.

OLs UserApp.java:handleClientRequest processes the UDP packet by the type of request. In both cases, OL sends a DIPLOMA DSM atom request, along with the additional information from the UDP packet, to the Remote Leader (RL). In the UploadPhoto case, RL is the same as the OL, since new photos are processed locally. In the DownloadPhoto case, RL is the leader of the target region. RLs UserApp.java:handleDSMRequest processes the DSM atom request from the OL. For UploadPhoto, the photo info is saved in RLs DIPLOMA memory as the first element of an ArrayList. (For the experiment, we only save one photo at a time. But theoretically, there is no limit to the number of photos that can be saved.) The reverse happens for DownloadPhoto, where RL retrieves the newest photo from its DIPLOMA memory. In both cases, RL sends a reply back to the OL, arriving at OLs UserApp.java:handleDSMReply which sends a UDP packet containing DIPLOMA latency and a success boolean, and also the photo data in the case of DownloadPhoto, back to the original non-leader.

Finally back to the original phone, its StatusActivity.java handler gets the UDP reply from OL and logs the replies. In the case of DownloadPhoto, the remote regions newest photo is displayed.

* I dont think the UI of leader showing new photos from nonleaders is important to write about

Leader transitions (its covered by other parts of the paper, right?) ———————— * leader go out of region TODO: add handing-off state? * leader is dead (cloud wait 100 secs to allow new leader) * leader heartbeat to the cloud and to non-leaders

Other things ———————— To avoid confusion and inconsistency of the region numbers, phone buttons only work if the phone is in a LEADER or NONLEADER state. To avoid double-sending a request (and possibly crash the camera surface view), a

ProgressDialog is shown until the client has received a leader reply or until a timeout.

Chapter 5

CameraCL Android Application

In CloudCamera, every request is sent to the cloud server. The cloud server keeps a dictionary linking each region to its newest photo. Codewise, CloudCamera only has one file: CameraCloud.java that is analogous to DiplomaCameras StatusActivity.java, but instead of sending UDP packets, CloudCamera sends HTTP post requests.

Chapter 6

Experiments

6.1 Experiment 1

6.2 Experiment 2

6.3 Experiment 3

6.4 Experiment 4

6.5 Experiment 5

6.6 Experiment 6

Chapter 7

Discussion and Conclusion

Bibliography