INSTITUTO SUPERIOR TÉCNICO

Departamento de Engenharia Informática

Project Assignment for Mobile Computing

MEIC/METI 2014-2015 – 2nd Semester

AirDesk

1 Introduction

The goal of this project is to develop a fully distributed mobile application named AirDesk, which aims at enabling content sharing and collaborative work within opportunistic mobile networks. Such networks will be established between the mobile devices of co-located users by leveraging the ad-hoc wireless communication capability of the devices. AirDesk will be targeted to Android version >= 4.0 and use WiFi Direct for wireless communication.

This project will give you the opportunity to acquire several skills in the field of mobile computing, namely the ability to: (a) specify the architecture of a mobile application based on a list of requirements, (b) implement mobile applications based on the Android's component paradigm, (c) manage wireless networks of mobile devices using WiFi Direct, (d) handle state replication and consistency in opportunistic mobile networks, (e) develop adaptive techniques to improve resource utilization, data availability, and performance, and (f) develop security mechanisms for mobile applications.

2 Specification

2.1 Basic Operation Mode

AirDesk aims to provide a common platform for collaborative work by co-located users. Specifically, AirDesk will enable users to connect wirelessly with nearby devices, and allow its users to grant each other's access to files.

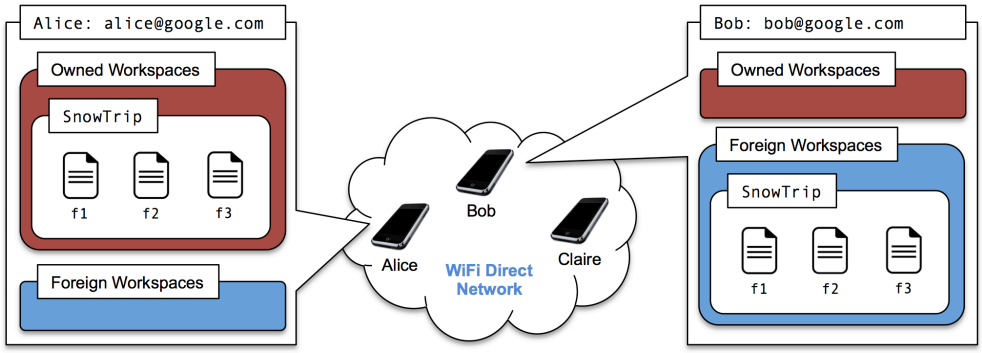

Figure 1: Simplified model of an AirDesk network enabling three users to share a workspace named "SnowTrip". Alice is the owner of the workspace; Bob and Claire are clients.

Figure 1 illustrates the file-sharing model to be implemented. In AirDesk, users can create and maintain *workspaces*. A workspace can be seen as a virtual folder for storing digital content, namely text files. A workspace can be shared by a group of users that belong in the same wireless network. A workspace features an access list of users with certain privileges: the *owner* and the *clients*. The owner is fully privileged: controls the lifecycle of the workspace, manages the access list membership, and has access to the workspace files. The clients are less privileged: they have some control of their access list membership, and have access to the workspace files. All users in the access list can see and edit the files of the workspace when they join the same network. For each user, AirDesk

shows two perspectives: one with the workspaces that he owns (*owned workspaces*), and another showing the workspaces that he is client of (*foreign workspaces*). Figure 1 illustrates this concept for a group of three users: Alice, Bob, and Claire. Alice is the owner of a workspace named "SnowTrip" and has shared it with two clients: Bob and Claire. In the devices of Bob and Claire, this workspace is mounted in their respective foreign workspace perspective, whereas in Alice's device "SnowTrip" is mounted in the owned workspace perspective. Users can then open the workspace locally and cooperatively plan their vacations in the mountains.

Workspace lifecycle and visibility

The lifecycle of a workspace it entirely determined by its owner. The owner defines when the worskpace is created, when it is destroyed, and who can access its files. To create a workspace, the owner must provide a meaningful name for it and define its maximum storage capacity (quota). The workspace name will then appear in the owned workspace perspective and files can be added to it. When the owner deletes the workspace, AirDesk removes its name from the owned workspace perspective and releases all resources allocated to the workspace.

In a client's side, the workspace is *mounted* whenever the client and owner belong in the same wireless network. Mounting the workspace means that its name appears in the client's foreign perspective and the files become locally visible and accessible to the user. If one of the owner or client leaves the network, the workspace is *unmounted*: its name is removed from the client's foreign workspace perspective and the files become inaccessible to the user. Unless the owner removes the client from the access list, the workspace will be automatically mounted if client join the owner's network in the future. However, the workspace reference will be definitely removed from the client's side if the workspace is deleted.

Access list management

The access list of a workspace defines which users have access permissions to the workspace. Naturally, the owner is always present in the list. Adding a client can be done in two ways:

a) By invitation: In this case, the workspace owner explicitly registers the ID of the client to the workspace's access list. AirDesk will automatically mount the workspace in the foreign workspace perspective of the targeted client. As stated previously, the mount operation is performed only when the device is in the same wireless network. To avoid naming conflicts, the user IDs will be email addresses. To facilitate group management and testing, a user's email will be associated with a short nickname. The email and nickname must be provided to AirDesk the first time the application is used.

b) By subscription: In this case, the initiative to join the workspace is taken by the client. For this to be possible, the workspace owner must flag the workspace to be *public* and write a public profile for the workspace, i.e., a set of meaningful tags that characterize the workspace content. Users can subscribe to public workspaces by providing a query, which consist of a set of keywords to be compared against the workspaces' public profiles. A match occurs if there is at least one keyword of the query that is equal to a tag in the profile. If a match occurs, the workspace is automatically mounted in his foreign workspace perspective. The workspace's access list is updated and the owner can see the ID of the newly added client. The owner does not have to explicitly accept the request. A user can subscribe to a public profile at any point in time. The subscription is validated every time a user enters or leaves a network. The client can edit the subscription to add or remove keywords from the query. Such changes may imply a reassessment of the workspaces to be mounted in foreign workspace perspective. A workspace owner can decide not to make his workspace public, by flagging it *private*. A private workspace will not be queried for profile match. If a workspace is marked private after being a public, the current access list is maintained: if the owner wants to exclude clients that were added by subscription, he must remove them from the list manually.

A client can be removed from the access list of a workspace by the owner or by the client himself. In this case, his access permissions are revoked and the workspace is unmounted from the client's foreign workspace perspective.

File access

At any given time, users can browse the workspaces available to them on AirDesk and access workspace files. The users of a workspace – both owner and clients – must be able to perform four operations to such files: *read*, *write*, *create*, and *delete*. Read file will show the file contents to the user on the screen. Write file will enable the user to

edit a file and modify its contents. Create file will create a new (empty) file in the workspace. Delete file will remove a file from the workspace. Note that file editing can be done directly in an AirDesk's panel; it is not necessary to open an external application for that. To simplify this operation, assume that all files are text files.

AirDesk must provide for the correct synchronization of a workspace's files. If a user performs a read operation, AirDesk will show the most recent version of the file. If a user tries to edit the file, AirDesk must first check whether or not someone else is already editing the file, otherwise the operation will be aborted in order to avoid conflicts. This also means that synchronization will be performed at the file-level granularity: no two users can edit a file simultaneously. When an editing operation is in course, the remote users observe the changes to the file only when the editor client commits the changes (e.g., presses a submit button). While the user is editing the file, the changes are only visible to him locally; in the meanwhile, if another user reads the file, he will see the original uncommitted file version. If a user creates a file, the operation must check for name conflicts and prevent the creation of two files with the same name. If a user deletes a file, the file will be removed from all of the workspace's perspectives, both owned and foreign. If a user is reading or editing a file and a delete operation was issued, the user will be notified and the ongoing read / edit operation will be canceled.

Workspace quotas

Workspaces have quotas, which define the workspace's maximum storage capacity. A write or a create operation that exceeds the workspace quota will fail. The quota value is defined upon workspace creation and can later be modified by the workspace owner. The upper quota limit is bound by the storage capacity of the device. The lower quota limit is conditioned by the storage space already taken up by the workspace: the user cannot set a smaller quota than the total size of the files presently stored in the workspace; to do so, files must be deleted first.

2.2 Advanced Mode

In addition to the basic operation described in the previous section, to attain the maximum score in the project, you are required to enhance your app with **one** advanced feature of your choice. We give you some suggestions:

- **A1. Server-backed storage.** In the basic operation, the files are created and maintained locally on the users' mobile devices. The goal of this feature A1 is to enable files to be stored on and loaded from the cloud. Several architectures are possible, for example by relying on storage services such as DropBox or Google Drive or on a dedicated server hosted in the computing clusters of Amazon EC2. Because the storage space in the cloud is much larger than the storage capacity of mobile devices, you need to develop clever hoarding techniques to improve file availability on the devices, reduce file access latency, and ensure data consistency. In particular, decisions must be taken as to which files are the most relevant to transfer between the cloud and the device, and when that file transfer should be carried out.
- **A2. Offline operation.** In the basic operation, the files of a foreign workspace are available only when the client is online, i.e., in the network of that workspace's owner. If one of them leaves the network, the workspace is unmounted from the client's foreign workspace perspective and the client is denied access to those files. The goal of this advanced feature A2 is to allow a client to keep accessing such files when operating off-line. To this end, the client endpoint must maintain local replicas of the foreign workspace files and make them available to the user when the device goes off-line. If you then allow for the modification of files by the clients, conflicts may arise if different users modify the same file independently. Therefore, you need to develop additional mechanisms to detect such conflicts and ensure consistency when the devices reconnect. To reconcile conflicting modifications, manual user intervention may be necessary.
- **A3. Security.** The basic operation provides no security at all. First, the access control permissions to files are coarse grained, giving clients only all-or-nothing access to all files of a workspace. Second, a malicious attacker can easily impersonate a user's ID by registering with the ID of that user. Furthermore, he can easily read or modify the content of workspace files by eavesdropping the network. This advanced feature A3 aims to improve the security of AirDesk. Several security extensions can be considered: (a) develop stronger user authentication mechanisms, (b) cryptographically protect the confidentiality and integrity of files, and (c) support fine-grained access control policies.

3 Development Phases

The project must be implemented progressively in three versions:

1 – Standalone Version (S-Version): This version must implement the basic functionality of AirDesk in which the user does not yet share workspaces with other users. For now, you do not need to manage the network: assume that the device will operate off-line. For this version, AirDesk must allow the user to: (a) maintain locally owned workspaces, (b) manage the files of his workspaces, and (c) set up quota restrictions. This version must also implement a perspective for the foreign workspaces. Since networking is not supported, to test this feature, it must be possible to mount the user's own workspaces in the foreign workspace perspective, thereby mirroring them in both perspectives. File operations in the foreign workspace perspective must be reflected in the owned workspace perspective and vice versa. To implement mirroring, simply allow for adding (by invitation) the user's ID to the access list of the workspace.

2 – Networked Version (N-Version): This version extends the S-Version with all network-dependent features: (a) management of WiFi Direct networks, (b) access list management by invitation and by subscription, and (c) full-featured multiuser workspace sharing. It will be necessary to implement distributed protocols, e.g., for state synchronization. The N-Version must implement all features of the basic operation mode (see Section 2.1).

3 – Advanced Version (A-Version): This version enhances the N-Version with **one** advanced feature (among those described in Section 2.2). You can freely choose which advanced feature to implement. Be creative: pick one idea from Section 2.2

4 Implementation

The target platform for AirDesk is Android >= 4.0. The application code must be written in Java. Android APIs can be used freely. The adoption of third-party libraries is disallowed unless explicitly approved by the faculty. AirDesk can be tested on software emulators or on real devices. Note, however, that if you choose real devices, the faculty will not be responsible for provisioning the hardware or for providing debugging support.

Networking

AirDesk nodes will communicate using the WiFi Direct wireless technology. WiFi Direct enables mobile devices to form ad-hoc networks as shown in Figure 2. In a WiFi Direct network, named P2P Group, one of the nodes – the Group Owner (GO) – is elected to play the typical role of a standard WiFi Access Point. The others nodes play the role of clients, who first need to obtain an IP address from the GO node to be able to communicate with other nodes in the network. Beware not get confused by the terminology! In WiFi Direct, the terms "owner" and "client" carry different meanings than within the scope of AirDesk's workspaces. WiFi Direct focuses on the network layer and below, whereas AirDesk on the application layer.

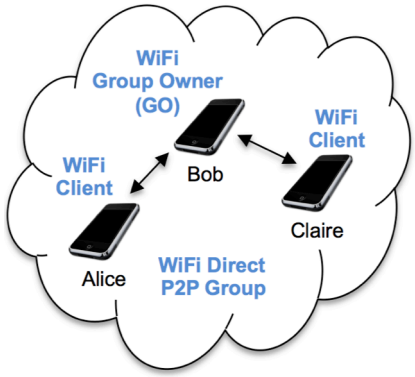

Figure 2: WiFi Direct network with three nodes: Bob is the GO node, Alice and Claire the client nodes.

Managing WiFi Direct networks is challenging because of its intrinsic dynamism. Given that the network nodes can move freely, you must take care to properly maintain the group membership and tolerate connectivity loss. AirDesk must gracefully support these cases:

a) **Network formation**: initial establishment of a network with at least two nodes: one GO and one client.

b) **Client churn**: a client enters or client leaves the network. A client may re-enter the network. The GO remains unchanged.

c) **GO churn**: the GO node leaves the network. AirDesk must provide for seamless handover and elect another node as GO to preserve connectivity among the remaining nodes.

You can test your prototype on their mobile devices or on Android emulator instances. To test your project on real devices, use the Android WiFi Direct API. To use the Android emulator, you will need WDSim, a toolkit that emulates WiFi Direct networks. WDSim will be provided in the website.

Tips for designing, testing, and debugging the application

Before starting to implement the project, study the requirements of AirDesk, and draw an activity wireframe. Make sure to design your application in a modular fashion. We suggest that you structure your application in well-defined modules by separating the presentation code (user interface), from the business domain code (e.g., workspaces), from the infrastructure code (networking, storage, etc.). Specify the interfaces to these modules allowing for their independent development and replacement. To help testing and debugging, we suggest that you implement two additional features: (a) a widget that lets the user explicitly disconnect the device from the network – this will be useful for testing node churn, (b) a widget to automatically populate a workspace with one or more files containing bogus data with total size defined by the user – this will allow you to test the quota limits, and (c) a way to pre-load workspace information and content into the application – this will avoid the need to initialize the application from scratch. To create unit tests that mimic user interaction with the application, we suggest the Robotium framework.

5 Grading Process

The projects will be evaluated based on several criteria. The most important aspects are: the functionality implemented, robustness and modularity of the implementation, technical quality of algorithms and protocols, and resource efficiency decisions. We will also assess the responsiveness and intuitiveness of the application interface. However, the esthetics of the GUI will **not** be considered! Therefore, keep the GUI as simple as possible.

The grading process comprises four stages:

- **April 10: Submission of Part I** –Part I corresponds to a fully functional prototype of AirDesk S-Version (see Section 3). The prototype sources will be submitted in the course website. Additional details on what is expected to be shown will be made available on the website. The Part I will account for 40% of the project's baseline grade. This submission is mandatory. Failing this milestone will result in the loss of the corresponding grade percentage. If you fail this submission, and only in that case, you can use an official prototype of Project Part I that will be made available by the faculty after the submission is due.
- **May 15: Submission of Part II** –Part II includes: a fully functional prototype of AirDesk versions N and A (see Section 3), and a final report. The prototype sources and the report must be submitted on the website. A template of the report will be published on the website. The report must describe what was and was not implemented, and it is limited to 5 pages (excluding cover). The cover must indicate the group number, and name and number of each of the group's elements. The Project Part II will account for 60% of the overall baseline grade, and is broken down into 40% for N-Mode, 15% for A-Mode, and 5% for the report.
- **May 18-22: Demonstration of the Project** – Each group will have 20 minutes to present the developed prototypes of AirDesk. Part I and Part II will be demonstrated independently. Each group must design the set of experiments to be presented in the demonstration. These experiments must be carefully prepared in order to showcase that the application works as expected.

  May 25-29: Discussion of the Project – The discussion will take 30 minutes per group. The baseline grade for the discussion will be weighed as shown in the formula below. However, the final grade awarded to each student will depend on his performance in the oral discussion and may vary within each group.

Project Baseline Grade = Part I x 40% + Part II x 60%

Good Luck!