MOBILE APPLICATIONS AND CLOUD COMPUTING

Roberto Beraldi
Course Outline

- 6 CFUs
- http://www.dis.uniroma1.it/~beraldi/MACC_18/index.html
- Topics:
  - Mobile application programming
    - Android
    - some comparison with iOS
  - Cloud computing
    - used as a backend for mobile apps
- In order to pass the exam, students must:
  - Write an individual working and documented application in android.
    - About 70% of the final score
  - Answer to 3 general questions
    - They are a necessary condition (it is a flag)
Key terms

- Mobile Application →
  - An app running on a mobile device → smartphone

- Cloud computing →
  - Delivery of remote virtual resources through internet (virtual machines, storage space, software functions (web api), data – e.g. open data, etc..)
Main platforms

- **Android (Google)**
  - Android Open Source Project (AOSP)
  - New versions lead by Google
    - Producers customize the new versions to their device
      - May take a while
    - Many marketplaces

- **iOS (Apple)**
  - Single official source
  - Single marketplace
Fragmentation

Android

<table>
<thead>
<tr>
<th>Version</th>
<th>Codename</th>
<th>API</th>
<th>Distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.3-2.3.7</td>
<td>Gingerbread</td>
<td>10</td>
<td>0.6%</td>
</tr>
<tr>
<td>4.0.3-4.0.4</td>
<td>Ice Cream Sandwich</td>
<td>15</td>
<td>0.6%</td>
</tr>
<tr>
<td>4.1.x</td>
<td>Jelly Bean</td>
<td>16</td>
<td>2.4%</td>
</tr>
<tr>
<td>4.2.x</td>
<td></td>
<td>17</td>
<td>3.5%</td>
</tr>
<tr>
<td>4.3</td>
<td></td>
<td>18</td>
<td>1.0%</td>
</tr>
<tr>
<td>4.4</td>
<td>KitKat</td>
<td>19</td>
<td>15.1%</td>
</tr>
<tr>
<td>5.0</td>
<td>Lollipop</td>
<td>21</td>
<td>7.1%</td>
</tr>
<tr>
<td>5.1</td>
<td></td>
<td>22</td>
<td>21.7%</td>
</tr>
<tr>
<td>6.0</td>
<td>Marshmallow</td>
<td>23</td>
<td>32.2%</td>
</tr>
<tr>
<td>7.0</td>
<td>Nougat</td>
<td>24</td>
<td>14.2%</td>
</tr>
<tr>
<td>7.1</td>
<td></td>
<td>25</td>
<td>1.6%</td>
</tr>
</tbody>
</table>

8.0 ➔ Oreo

As measured by the App Store on September 6, 2017.

iOS11 ➔ 64 bit
Developing strategies

• Web site for mobile phones

• Web app

• Native apps on the mobile
  • Android (C, java, Kotlin)
  • iOS (Swift)
What is a mobile platform?

- An Operating System
- A middleware (set of frameworks or libraries that provide the a core set of functionalities)
- A set of apps
- A developer environment
Some key concept: Event-driven app

- A mobile app reacts to events generated from user actions (like pushing a button) making computation and showing a new screen.

- Code computing the visual reaction to an event runs inside a ‘main’ thread (so that the screen is quickly updated, maybe showing a progress bar).

- Slow operations runs inside working threads.
Some key concept: component-based apps

- A mobile application is a set of software components, implementing a known interface
- The middleware calls methods of this interface
- Software components have a lifecycle
  - `onCreate` (android)
  - `ViewController` (iOS)
Component-based apps

UI

SW components

Middleware

SW you write
MVC

- View
- Controller
- Model

- Seen by
- Uses
- Update
- Manipulates

[wikipedia]
MVC sw architecture

- Each screen displays a ‘view’
  - A view can be any output representation of information, such as a chart or a diagram.

- Model
  - Data are stored in a model, that is unaware of how data are presented in views. May notify the view when data changes

- Controller
  - Sits between view and model. It receives user inputs and converts it to commands for the model or view
Model View Presenter vs MVC
The three-tier architecture

- Presentation tier
- Business logic
- Data tier

- Mobile Backend (as a Service)
Mobile backend

- Apps may rely on a set of functionalities running on a remote server.
- These functionalities are accessed through Internet according to a precise protocol.
Some example

- Firebase
- GAE ...

- Google Maps

- Wheatear Underground
  - Dropbox
  - Flicker
  - Facebook (e.g., OAuth)
  ...

GUI

May be Strongly Integrated in IDE

Web API, WS
(may provide SDK)

Backend
Some example

- Cloud9
- Heroku
- PaaS & SaaS (write your own API)
- Web API, RESTful WS

Mashup App

Backend
Comparison with other architectures

• Monolithic application
  • All the logic is in a single program

• Web App
  • The server dynamically generates the page to be rendered with the response

• Mobile app with backend
  • A server provides primitive operations to an application that runs on a mobile device.
Monolithic architecture

All three tiers run inside the mobile device
Impossible to have too concurrent players

e.g., JavaScript, or any language
Web App (c/s arch 1)

- Client

http://game.org/ttt.php?move=7&ply=1

Presentation layer = browser

Presentation layer (generate html page)
Business layer (php program, e.g., check winner)
Data layer (store chess cons)
Web app (C/S app 2)

Client -> HTTP/HTML -> Server

HTTP

TCP/IP connection 3306,...

User Agent

mod_php

MySQL

PHP

XAMPP

WampServer

Ubuntu, Fedora, Debian
Mobile app + backend

- Send the current state
- Reply with the new state

Presentation (View) + Controller

Interface (remote functions)

Web API (cloud)

Business logic + data
Cloud computing and mobile applications

- Cloud computing itself is an umbrella term to refer to a way to access functionalities remotely (e.g., via Internet).

- There are different ‘delivery models’ of cloud computing:
  - **Mobile Backend as a Service**
  - **Software as a Service** (SaaS) or **Web API**
  - Platform as a Service (PaaS)
  - Infrastructure as a Service (IaaS) (virtualization)
  - Storage as a Service
  - ...

- Cloud computing, or more simply ‘cloud’, can be exploited by mobile apps.
Example: Dropbox

- Dropbox is a free (base 2GB, up to 18GB), web-based cloud storage mechanism
  - file backup service
  - data sharing
  - Data sync among different clients
  - Mobile applications

Windows

Linux

MAC

Web-API: REST / JSON calls

Freemium business model

Security (SSL, AES-256 bit), scalability (load balancer, server notification, data/metadata..)
Dropbox usage

• Two access «channels»
• For users:
  • Web based access
  • Proxy applications
• For developers:
  • Endpoint for web-api calls
    • Wire protocol
    • Data representation
  • Different development technologies
    • Java, .NET, Python, etc.
Example 2

- contacts
- photo
- messages
.. Phone backup

-iCloud drive
(store any kind of document)
Example 3: Amazon’s Cognito

https://aws.amazon.com/it/cognito/
Characteristic of a cloud storage service

• Bundling
  • When a batch of files is transferred, files could be *bundled*, so that transmission latency and control overhead are reduced.
    • For example, Dropbox use only few TCP connections to transfer multiple files

• Chunking
  • Large files can be either monolithically transmitted to the cloud or chunked into smaller pieces. Chunking is advantageous because it simplifies recovery in case of failures
    • Dropbox 4MB, Google Drive 8MB,…

• Compression
  • data compressed before a transfer. Compression could, in general, reduce traffic and storage requirements at the expense of local processing time.
Characteristic of a cloud storage service

• Deduplication
  • Server data deduplication eliminates replicas on the storage server.
  • This can be accomplished by calculating a file digest using the file content, e.g., SHA256 is used by Dropbox
  • The digest is sent to servers prior to submitting the complete file.
  • Servers then check whether the digest is already stored in the system and skip the upload of repeated content.
  • Dropbox implements inter-user deduplication: this technique allows a user to skip submitting files that are already stored by any other user.
Characteristic of a cloud storage service

• Delta encoding
  • Delta encoding calculates the difference among file revisions, allowing the transmission of only the modified portions.
  • Indeed, delta encoding provides similar benefits as the combination of chunking and deduplication, but with a finer granularity.
Comparison among personal cloud storage(*)

<table>
<thead>
<tr>
<th>Service</th>
<th>Bundling&lt;sup&gt;1,2&lt;/sup&gt;</th>
<th>Chunking&lt;sup&gt;2&lt;/sup&gt;</th>
<th>Compression</th>
<th>Deduplication</th>
<th>Delta Encoding</th>
<th>P2P Sync</th>
</tr>
</thead>
<tbody>
<tr>
<td>Box</td>
<td>no (mc)</td>
<td>no</td>
<td>never</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Cloud Drive</td>
<td>no (up mc, down mt)</td>
<td>5 MB</td>
<td>never</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Copy</td>
<td>no (mt)</td>
<td>4 MB</td>
<td>always</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Dropbox</td>
<td>yes</td>
<td></td>
<td>smart</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>Google Drive</td>
<td>no (mc)</td>
<td>8 MB (down only)</td>
<td>smart</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>hubiC</td>
<td>no (mt)</td>
<td>no</td>
<td>never</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Mega</td>
<td>no (mc)</td>
<td>1 MB</td>
<td>never</td>
<td>partially</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>OneDrive</td>
<td>no (mt)</td>
<td>up 4 MB, down 1 MB</td>
<td>never</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Wuala</td>
<td>no (mt)</td>
<td>4 MB</td>
<td>never</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>ownCloud</td>
<td>no (mt)</td>
<td>no</td>
<td>smart / down only</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>Horizon</td>
<td>no (mt)</td>
<td>1 MB (up only)</td>
<td>never</td>
<td>partially</td>
<td>no</td>
<td>no</td>
</tr>
</tbody>
</table>

<sup>1</sup> mc: one or multiple TCP connections per file; mt: one or multiple application layer transactions per file.

<sup>2</sup> up and down refer to the upload and the download experiments, respectively.

Example 4: Microsoft’s Intune

Microsoft Intune helps organizations provide their employees with access to corporate applications, data, and resources from virtually anywhere on almost any device, while helping to keep corporate information secure.

Intune can manage:
- **Mobile devices** (including phones and tablets running Android, iOS, Windows Phone and Windows RT operating systems).
- **Computers** running Windows
Example 5: Firebase (push notification)

GCM allows to attach up to 1,000 recipients to a single message.

Firebase Cloud Messaging (FCM) is the new version of GCM.
Example 6: Amazon’s Simple Notification Service (SNS)

• It is a fast, flexible, fully managed push notification service that lets you send individual messages or to fan-out messages to large numbers of recipients.

• Amazon SNS makes it simple and cost effective to send push notifications to mobile device users, email recipients or even send messages to other distributed services.
Example 6: Amazon’s Simple Notification Service (SNS)

- Seamlessly scale from a handful of messages per day to millions of messages or higher.

- With SNS you can publish a message once, and deliver it one or more times. So you can choose to direct unique messages to individual Apple, Google or Amazon devices, or broadcast deliveries to many mobile devices with a single publish request.

- SNS allows you to group multiple recipients using **topics**.

- A topic is an “access point” for allowing recipients to dynamically subscribe for identical copies of the same notification.

- When you publish once to a topic, SNS delivers appropriately formatted copies of your message to each subscriber.
A Web API allows to design and develop an application that exploits software modules accessed via Internet via a simple wire protocol.
Web API: example
Web API: programmatic point of view

Web URL (standard HTTP methods)

HTTP

web browser

Rest, XML-RPC, etc

HTTP

Application

Programmatic Service Access (many methods)
Web API

- Many players do offer services via a web api
  
  ProgrammableWeb

- Design and develop a web-api
  - PHP
  - ASP.NET
  - Ruby on Rails

  ....
Web API and Interoperability

A=Android
B=iOS
C=Windows 10
Example: Google apis

<table>
<thead>
<tr>
<th>Service</th>
<th>Status</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AdSense Management API</td>
<td>OFF</td>
<td>Courtesy limit: 10,000 queries/day</td>
</tr>
<tr>
<td>Analytics API</td>
<td>OFF</td>
<td>Courtesy limit: 50,000 queries/day</td>
</tr>
<tr>
<td>Audit API</td>
<td>OFF</td>
<td>Courtesy limit: 10,000 queries/day</td>
</tr>
<tr>
<td>Blogger API</td>
<td></td>
<td>Request access...</td>
</tr>
<tr>
<td>Books API</td>
<td>OFF</td>
<td>Courtesy limit: 1,000 queries/day</td>
</tr>
<tr>
<td>Calendar API</td>
<td>OFF</td>
<td>Courtesy limit: 10,000 queries/day</td>
</tr>
<tr>
<td>Custom Search API</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Freebase API</td>
<td>OFF</td>
<td></td>
</tr>
<tr>
<td>Google Affiliate Network API</td>
<td>OFF</td>
<td>Courtesy limit: 1,000 queries/day</td>
</tr>
</tbody>
</table>
Mashup applications

CLIENT

API WEB

Provider A

Provider B

Provider ...

s=A.methodName(par₁)
...
z=B.methodName(s,par₂)
Mashup applications (1/2)
Mashup applications (2/2)

HTTP - Server Web
- "mashupper"
  (e.g., dynamic pages Php)

WEB API

XML-RPC

SOAP

REST
Some idea about the future (not studied in this course)

Is it “strange”?

D2D communications
(like, p2p, or multi-hop)
Some idea about the future (not studied in this course)

FANET
Some idea about the future (not studied in this course)

- Code offloading..
Some idea about the future \textit{(not studied in this course)}

IoT and Fog Computing
The term fog computing, coined by Cisco, refers to the need for bringing the advantages and power of cloud computing closer to where the data is being generated and acted upon. Fog computing reduces the amount of data that is transferred to the cloud for processing and analysis, while also improving security. Well suited for Internet of Things.
• QUESTIONS?
WARNING

- Projects can be done in groups (2-3 max)
- Try to use GitHub
How to create a mobile app?

- To develop a mobile app, one can use methodologies from software engineering.
- The ‘agile’ methodology is one possible well established method.
  - Many agile frameworks exist, e.g., Scrum.
- We will shortly see the design and development strategy.
- The entire framework is studied in SE.
Behaviour Driven Design, aka BDD

- Elicit functional requirements through **user stories**
- Elicit UI requirements through a **storyboard**
- Use paper and pencil to sketch screens of the UI (!)
- Or… write a mockup (code that produces the screens)
- Many specialized tools are available
  - For example: [https://marvelapp.com](https://marvelapp.com)
  - You may (even) try to use power point ..

  [https://marvelapp.com/project/2294932/](https://marvelapp.com/project/2294932/)
Example
Example

You are logged in!
Example
User stories

• User stories are a set of short descriptions about what a user can do with the application
  • An application is composed of many US

• A single user story is a narrative description of a single feature. It has the following structure

  • As a <role>, I want <desire> so that <benefit/goal>
User stories (example)

- As a user, I **want** to change the color used by the current drawing tool **so that** it only affects the next use of that tool

- As a movie fan, I **want** to add a movie to the current list of movies, **so that** I can share comments with other fans
User stories and screens

- A user story should be small enough to be represented by a single screen
Storyboard

• Elicit UI users requirements from User Stories
• Each UI view is a sketch (maybe obtained with paper and pen)
• **Storyboard** is a tree or graph capturing the relationship between sketches according to the user inputs
• In a mobile app it generates the navigation flow
App’s navigation flow

- Set of screens and their relationship

Android has a back button
A complete example

- Goal of the application: share the parking place of a family car, so that each user can know the position of the car
User stories (example)

• US1: I want to login in the app, so that only authorized users can use the app

• US2: As an authenticated user, I want to store the position of the car on a map, so that another user can later locate the car on the map

• US3: As an authenticated user, I want to know the position of the current car, so that I can pick it

• US4: As an authenticated user, I want to set the state of the car as busy or available, so that other users can know its current state
Mapping US1

- US1: I want to login in the app, so that only authorized users can use the app
Mapping US2

• US2: As an authenticated user, I want to store the position of the car on a map, so that another user can later locate the car on the map
• **US3**: As an authenticated user, I want to know the position of the current car, so that I can pick it
• US4: As an authenticated user, I want to set the state of the car as busy or available, so that other users can know its current state
Storyboard (1/2)

Login

Park here

park

Some connection is missed..
Storyboard (2/2)

Select a function

Login

Select
- Park here
- Status

Set status

park

ERROR

busy
Storyboard
Storyboard

LOGIN ERROR!
Storyboard

Select

- Park here
- Status

Park here
Storyboard
Storyboard
User navigation flow
Developing strategy

- Developing time (Test Driven Development, TDD)
Example: Unit test

- Unit Test: the smallest test can be run against a portion of sw (like a method in a class). It can fail (red), not yet be implemented (orange), pass (green).

- It measure different metrics, like code coverage, etc..

- In TDD, Unit Tests are written before the code
  - The test of course initially fails, but as code is added the color changes from red to green

- When several ‘modules’ are developed, they are integrated tested
Example: testing in android – Unit Test

- Used when your there are no Android framework dependencies (or when you can mock the Android framework dependencies, see later)
- Test a small piece of code (e.g., a method of a class)
  
  1. Initialize a small piece of the code to test
  2. Apply stimulus (i.e., calling methods)
  3. Compare the result with the expected one
Example: testing in adroid Junit

```java
import org.junit.Test;
import java.util.regex.Pattern;
import static org.junit.Assert.assertFalse;
import static org.junit.Assert.assertTrue;

public class EmailValidatorTest {
    @Test
    public void emailValidator_CorrectEmailSimple_ReturnsTrue() {
        assertEquals(EmailValidator.isValidEmail("name@email.com"), true);
    }
    ...
}
```
Example: testing in android
Example: Unit test in android - mocking

- Mock objects do the mocking of the real service. A mock object returns a dummy data corresponding to some dummy input passed to it.
- In android: Mockito
Testing

• Testing is an important and fundamental part of any application
• In our lecture we will see some simple example
• For more info see web resources:
  • Android:
  • https://developer.android.com/training/testing/fundamentals.html
  • iOS:
  • https://developer.apple.com/library/content/samplecode/UnitTests/Introduction/Intro.html#//apple_ref/doc/uid/DTS40011742-Intro-DontLinkElementID_2
Connecting app to the cloud

Login

Login with Google, FB, ..

Google map

Car position
Connecting mobile app to the cloud

Standard library (HTTP Client)

HTTP Server + Router

CLOUD SERVICE
Connecting mobile app to the cloud

![Diagram showing the connection between a mobile app and the cloud through a plug-in (proxy) and HTTP server/router.](image)
Questions?