Progettazione e Sviluppo di un'Applicazione Mobile di Marketing ed E-Commerce

Tommaso Berlose

2016-11

Indice

Intro

Provaaaaaaaaaaaa

0.1 Situazione esistente

0.1.1 App Economy

Figura 1: App Economy

Non vi è forse modo di descrivere la società attuale e questo periodo storico senza valutare l'importanza dello sviluppo tecnologico che ci ha portato nell'*Era dell'Informazione*. La rivoluzione tecnologica che sta avvenendo in questi anni, specialmente a partire dagli Novanta, ha portato la connessione globale ad internet ad assumere un ruolo essenziale in ogni aspetto della società moderna e della nostra vita.

È cambiato drasticamente il modo con cui le persone accedono alle informazioni, che siano queste di tipo personale o di tipo economico. Allo stesso modo si sono dovute adeguare le strategie di tutte quelle aziende che hanno visto cambiare in maniera drastica il proprio mercato, invaso molto volte da tecnologie sempre diversificate e innovative.

Secondo Paul Mason, è stata la dottrina economica degli ultimi decenni della nostra epoca che, da una parte ha avuto il merito di promuovere la più grande ondata di sviluppo economico che il mondo abbia mai visto, ma dall'altra ha portato a mercati incontrollatie a vorticosi cambiamenti sociali innescati dalla tecnologia. Allo stesso modo si sono diffusi concetti come i progetti open-source, la sharing economy o le licenze creative commons che hanno portato ad uno stravolgimen-

to di molti mercati dove tante aziende hanno dovuto rivedere la propria business strategy.[1]

Primi su tutti sono diventati di fondamentale importanza i concetti di e-commerce e marketing digitale, fortemente spinti dalle tecnologie e dalla nuova possibilità di accedere ad una risorsa comune (internet) da parte della maggior parte della persone anche di cultura, età e ambienti sociali diversi. Diventa quindi fondamentale l'Application Economy: lo sviluppo e l'utilizzo di applicazioni mobile multipiattaforma per raggiungere gli utenti consumatori. Questa strategia può essere applicata in ambito di marketing per pubblicizzare un proprio prodotto, fidelizzare il consumatore alla propria azienda e mantenere una propria immagine in un sistema in cui l'idea che i consumatori hanno dell'azienda è fondamentale.

Forse ancora in via di sviluppo ma ormai più diffuso è l'e-commerce, la possibilità di fare acquisti online, che ormai sta lentamente, ma inesorabilmente, spostandosi sui device mobile. Proprio in questi ultimi anni tutti i principali protagonisti del mondo software hanno presentato la propria visione delle vendite online, primi fra tutti Amazon. A cui ovviamente si stanno affiancando i sistemi di pagamento online tramite device mobile: paypal, android pay, apple pay, ecc...

0.1.2 Ecosistema MyGelato

L'ecosistema MyGelato si pone proprio all'interno di questo contesto, ad unire le strategie di marketing dell'azienda produttrice Carpgiani e delle singole gelaterie convenzionate; insieme alla possibilità di comprare, regalare e utilizzare dei coupon che permetto l'acquisto online di gelati.

È un progetto di piattaforma web e mobile che si rivolge al pubblico dei gelatieri e dei consumatori di gelato. Diversi sono gli obiettivi commerciali di una piattaforma come MyGelato, tutti aventi come fine ultimo l'aumento del fatturato. Tramite l'E-commerce: con lo scopo di incentivare la compravendita di gelati tramite un sistema di coupons digitali e il Marketing: è necessario promuovere il consumo di gelato tramite offerte e fornire ai gelatieri un sistema semplice ed efficace per pubblicizzare il proprio negozio.

0.1.3 Stato dell'Arte dell'Applicazione Android

[1] Paul Mason. *Postcapitalismo.* 2016.

0.2 Specifiche di Progetto

0.2.1 Rifare app uguale

0.2.2 temi con flavors

0.2.3 risorse da adattare

0.2.4 parallelismo con iOS

0.2.5 multilingua, login facebook, ecc...

0.3 Scelte di Progetto

Definiti gli obiettivi progettuali dell'applicazione si sono valutati gli strumenti di sviluppo da utilizzare durante lo svolgimento della tesi: come parametri si é tenuto conto di tempistiche di aggiornamento, adeguamento alle linee guida del sistema operativo in oggetto, documentazione disponibile e modernità delle tecnlogie utilizzate.

Durante la prima fase di sviluppo strutturale si è dovuto valutare la tecnologia da utilizzare in modo da tenere il passo con le ultime specifiche di Android e tenendo conto che, anche secondo gli ultimi report pubblicati da Google, rimane una forte frammentazione della distribuzione del sistema operativo, dovuta ai molteplici produttori di hardware.[2]

Era quindi da tenere in considerazione la retrocompatibilità delle librerie utilizzate, avendo scelto di supportare fino alla versione 16 del SDK (Android 4.1 Jelly Bean), e la possibilità di utilizzare l'applicazione anche su device con schermi e hardware differenti.

Si sono inoltre adottate alcune strategie implementative, come la creazione di custom view, per poter meglio strutturare il progetto favorendone anche eventuali modifiche o ampliamenti.

0.3.1 Strumenti di Sviluppo

Considerato che l'applicazione mobile, oggetto di tesi, era destinata a lavorare unicamente sulla piattaforma Android, si è scelto di utilizzare come principale strumento di sviluppo **Android Studio**, software dedicato alla programmazione nativa del suddetto sistema operativo.[3]

Durante lo sviluppo del progetto si è utilizzato come sistema di versioning il software **Git**, così da poter mantenere uno storico del lavoro svolto e ottenendo tutti i vantaggi del'utilizzo di un Sistema per il Controllo di Versione (*Version Control System - VCS*).

Sono inoltre da citare alcuni software utilizzati durante il lavoro di tesi che, anche se di minore impatto, sono serviti a testare e sviluppare alcune parti fondamentali di codice. Primo fra tanti la web application Postman[4] che permette di testare e sviluppare le APIs con cui due applicativi possono dialogare, in maniera semplice ed intuitiva. Utile per valutare le chiamate al server e per verificare le risposte ottenute. **Altri da citare???**

[2]Google Inc. *Android Developers*. URL: `https://developer.android.com/about/dashboards/index.html`.

[3]Google Inc. *Android Developers*. URL: `https://developer.android.com/training/basics/firstapp/index.html`.

[4]Inc. Postdot Technologies. *Postman*. URL: `https://www.getpostman.com/`.

Android Studio

Android Studio[5] è un ambiente di sviluppo integrato (IDE) per lo sviluppo per la piattaforma Android. È stato annunciato il 16 maggio 2013 in occasione della conferenza Google I/O e la prima build stabile fu rilasciata nel dicembre del 2014. Basato sul software della JetBrains IntelliJ IDEA, Android Studio è stato progettato specificamente per lo sviluppo di Android.[4] È disponibile il download su Windows, Mac OS X e Linux,[5][6] e sostituisce gli Android Development Tools (ADT) di Eclipse, diventando l' IDE primario di Google per lo sviluppo nativo di applicazioni Android. Permette di creare un progetto gradle, apk, github,

Git

Git è un software di controllo versione distribuito (*Distributed Version Control Systems - DVCS*) utilizzabile da interfaccia a riga di comando, creato da Linus Torvalds nel 2005.[6] Nacque per essere un semplice strumento per facilitare lo sviluppo del kernel Linux ed è diventato uno degli strumenti di controllo versione più diffusi.

Per quanto riguarda qualsiasi tipo di progetto di IT (*Information Technology*), specialmente se si tratta di un lavoro da dover svolgere in team, Git da la possibilità di mantenere in memoria tutte le versioni del proprio lavoro (sia in locale che online). Questo permette a più persone di poter accedere alla cronologia del lavoro condiviso, avendo anche la possibilità di verificare la presenza di errori e di eliminarli tornando ad una versione precedente del progetto. Git sfrutta alcuni algoritmi avanzati per calcolare le differenze riga per riga tra i file di diverse versioni così da verificare la presenza di errori o conflitti.

Git è fortemente direzionato verso uno sviluppo progettuale non lineare. Supporta diramazione e fusione (*branching and merging*) rapide e comode, e comprende strumenti specifici per visualizzare e navigare l'intero storico delle versioni anche se proposte da sistemi differenti. È veloce e scalabile nella gestione di grandi progetti ed è molto utile nello sviluppo in team, specialmente se affiancato dal modello GitFlow che permette di gestire rami specifici per il developing o per le release.

Infine grazie all'utilizzo di piccoli accorgimenti come il file .gitignore (che permette di non salvare anche i file temporanei e non importanti del progetto) e soluzioni online come GitHub o BitBucket (quest'ultimo utilizzato durante lo sviluppo di questa tesi) il processo di sviluppo software viene drasticamente avvantaggiato.

[5]Wikipedia. *Android Studio.* URL: `https://it.wikipedia.org/wiki/Android_Studio`.
[6]Wikipedia. *Git (software).* URL: `https://it.wikipedia.org/wiki/Git_(software)`.

0.3.2 Programmazione Nativa

Questa tesi si inserisce all'interno del lavoro di modifica e miglioramento delle applicazioni mobile dell'ecosistema MyGelato, il quale includeva un progetto sia per piattaforma Android sia per piattaforma iOS. Essendosi creati due differenti progetti, durante la prima fase di progettazione, si è potuto valutare in maniera più libera l'utilizzo o meno della programmazione nativa.

Sicuramente lavorare in questo modo o meno ha i propri pro e contro. Mentre da un lato il nativo offre la possibilità di una gestione totale del dispositivo senza la paura di trovare limiti, d'altra parte richiede spesso una programmazione molto professionale e si concentra esclusivamente su una piattaforma impedendo un'agile riciclo dei propri sforzi su altri mercati del mobile. Il non-nativo, invece, offre diversi vantaggi ascrivibili ad una minore necessità di programmare e molto spesso alla possibilità di creare applicazioni cross-platform distribuibili su sistemi operativi diversi.[7]

Non avendo quindi necessità di mantenere alta la portabilità del codice su altre piattaforme, si è preferito sviluppare tramite programmazione natuva; potendo quindi sfruttare pienamente l'architettura del sistema operativo sottostante.

Java

XML

0.3.3 Database

Realm

0.3.4 Chiamate Server

Attualmente ogni tipo di piattaforma che voglia permettere al proprio utente di accedere da remoto, e quindi da qualsiasi dispositivo, alle proprie informazioni ha come parte fondamentale lo sviluppo e il mantenimento di un server di beckend (quindi nascosto nelle funzionalità alla vista dell'utente) che mantenga, gestisca e restituisca i dati necessari alle funzioni degli applicativi. Nello stesso modo la piattaforma MyGelato si basa su un server che mantiene ogni informazione riguardante utenti, shop e cards fornendo delle API REST che vengono sfruttate dalle applicazioni mobile e web per poter fornire i propri servizi.

Il passaggio di informazioni tra i due applicativi avviene tramite chiamate internet con protocollo http protette grazie ad una connessione criptata dal Transport Layer Security (TLS) o dal suo predecessore, Secure Sockets Layer (SSL). Grazie

[7] HTML.it. *Come sviluppare app Android, ibrido o nativo?* URL: `http://www.html.it/pag/48523/alternative-allo-sviluppo-nativo/`.

alla API REST è possibile sviluppare in maniera strutturata e di seguire alcuni pattern delle tecnologie che permettono di valutare ogni possibile errore delle connessioni: errori nei dati, nell connessione, ecc... In Java le chiamate http sono abbastanza difficili da gestire poichè non sono automatizzate e son stati effettuati dei cambiamenti tra le ultime versioni che non permettono di lavorare correttamente su tutti i dispositivi Android. Per questo motivo è stata utilizzata la libreria open-source OkHttp sviluppata da Square, che permette di automatizzare le chiamate http sfruttando anche una notevole semplificazione del codice utilizzato per programmare.[8]

OkHttp

La liberia OkHttp nasce dal fatto che il protocollo http è l'attuale mezzo di comunicazione per ogni applicativo moderno. Per questo si pone l'obiettivo di semplificarne l'utilizzo in più di una tecnologia migliorandone le prestazioni e fornendo allo sviluppatore alcuni automatismi comodi durante lo sviluppo e la programmazione.

Supporta ogni tipo di comunicazione http, ne gestisce la latenza, il caching, la riconnessione, l'utilizzo di protocolli sicuri come TLS, l'utilizzo di indirizzi IP multipli ed è molto semplice da integrare e utilizzare all'interno del proprio software. Grazie ad un'interfaccia molto semplice permette di automatizzare le chiamate http utilizzando dei Builder che lasciano specificare solo i parametri necessari e restituiscono un unico oggetto da cui si ottiene la risposta, positiva o negativa, della richiesta.

```
OkHttpClient client = new OkHttpClient();
Request request = new Request.Builder()
            .url(endpoint)
            .addHeader("Accept","application/json")
            .build();

Response response = client.newCall(request).execute();
```

Inizializzato il client OkHttp basta creare un oggetto request a cui vanno passati i parametri necessari; basta poi eseguire la chiamata per ottenere un oggetto response all'interno del quale sono presenti i dati scaricati, il codice di risposta della chiamata e anche eventuali errori.

[8] Square. *OkHttp*. URL: `http://square.github.io/okhttp/`.

API REST

0.3.5 Gestore di Eventi

Un passaggio fondamentale durante lo sviluppo di un software è l'aggiornamento delle view a seguito di un cambiamento nello stato dell'applicazione. In Android è necessario intervenire direttamente sugli elementi dell'interfaccia modificando ogni oggetto in base alle specifiche richieste senza poter sfruttare veri e propri automatismi. Una forte limitazione del sistema operativo è dovuta, per esempio, al fatto che l'unico thread che ha il permesso di accedere e di modificare l'interfaccia utente è il Main Thread; questo complica l'interazione con eventuali sistemi multithreading anche se, ovviamente, fa parte delle specifiche di sistema utili a limitare i conflitti sull'accesso alle risorse condivise (in questo caso le view).

Questo problema necessita di ampia valutazione durante lo sviluppo e anche in questo caso si è dovuto strutturare l'applicazione in modo che non incorresse in errori di inconsistenza tra le view a cui il programma voleva accedere e le view realmente visualizzate al momento. Nel caso si utilizzino script asincroni è molto facile tentare di accedere ad elementi, o anche intere activity, non più presenti nell'interfaccia utente; per questo motivo si è scelto di utilizzare la libreria EventBus sviluppata da GreenRobot[9] e pensata esattamente per risolvere questo tipo di errori.

EventBus

EventBus è una libreria open-source per Android che utilizza il pattern publish/subscribe per far dialogare tutti i componenti di un'applicazione. Disaccoppia le classi che interagiscono sulla stessa interfaccia e le gestisce in maniera centralizzata monitorando le performance e gestendo gli errori che potrebbero essere sollevati da uno sviluppo multithreading.[10]

I principali benefici che si riscontrano tramite l'utilizzo di questa libreria sono la semplificazione delle comunicazioni tra componenti, il disaccoppiamento tra mittenti e riceventi degli eventi, miglioramenti nelle perfomance, facile integrazione dei metodi da utilizzare ed infine anche caratteristiche avanzate come priorità e indirizzamento a thread specifici.

Durante lo svolgimento di questa tesi si è definita una struttura abbastanza comune formata da: - AsyncTask, oggetto che permette di elaborare una serie di informazioni in background, utilizzato per eseguire il download delle informazioni dal server; - View specifica dell'activity con determinati componenti da aggiornare

[9] GitHub. *EventBus*. URL: `https://github.com/greenrobot/EventBus`.
[10] GreenRobot. *EventBus*. URL: `http://greenrobot.org/eventbus`.

in base allo stato dell'applicazione; - Metodo updateUI() che preso in gresso i dati scaricati doveva aggiornare le view rispetto allo stato.

Per far dialogare questi elementi si è utilizzato EventBus così da sfruttare il lavoro di thread in background senza limitare l'utente nell'attesa di un determinato evento. Si procedeva quindi crendo un evento specifico per l'interazione da dover gestire di cui qui viene riportato un esempio:

```
public static class UpdateCouponUiEvent<T> {
  public T data;
  public UpdateCouponUiEvent() {
  }
  public UpdateCouponUiEvent(T data) {
    this.data = data;
  }
}
```

EventBus permette di registrare dei metodi eseguendo una subscribe ad un determinato evento: ogni qual volta un evento di quel tipo viene sollevato, il metodo viene eseguito. La subscribe deve essere definita nel codice nel seguente modo:

```
@Subscribe
public void updateUI(UpdateCouponUiEvent event) {
  // update UI with event.data
}
```

Grazie all'utilizzo di eventi custom è dunque possibile passare anche dei dati tramite l'evento che viene lanciato, nel codice messo ad esempio si tratta della variable data, definito di tipo generico.

Infine all'interno del AsyncTask, a conclusione del download delle informazioni e dell'aggiornamento dello stato, era sollevato un evento grazie ad una publish:

```
EventBus.getDefault().post(new UpdateCouponUiEvent());
```

0.3.6 Notifiche Push

Geofencing

Firebase

0.3.7 E-Commerce

Stripe

0.3.8 Librerie Minori

0.3.9 Scelte progettuali strane e che non so come chiamare

Flavors' Custom View

0.4 Implementazione

0.4.1 Shop

Ricerca

Preferiti

0.4.2 Carte gelato

Gelato Master

Notifiche Push

0.4.3 Coupon

Acquisto

Regalo

Utilizzo e Validazione

Bibliografia

GitHub. *EventBus*. URL: https://github.com/greenrobot/EventBus.

GreenRobot. *EventBus*. URL: http://greenrobot.org/eventbus.

HTML.it. *Come sviluppare app Android, ibrido o nativo?* URL: http://www.html.it/pag/48523/alternative-allo-sviluppo-nativo/.

Inc., Google. *Android Developers*. URL: https://developer.android.com/about/dashboards/index.html.

— *Android Developers*. URL: https://developer.android.com/training/basics/firstapp/index.html.

Mason, Paul. *Postcapitalismo*. 2016.

Postdot Technologies, Inc. *Postman*. URL: https://www.getpostman.com/.

Square. *OkHttp*. URL: http://square.github.io/okhttp/.

Wikipedia. *Android Studio*. URL: https://it.wikipedia.org/wiki/Android_Studio.

— *Git (software)*. URL: https://it.wikipedia.org/wiki/Git_(software).

Elenco delle figure

0.5 Codice

0.6 Ringraziamenti