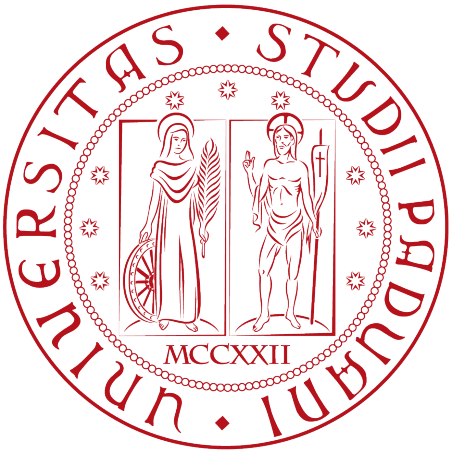
**Università degli Studi di Padova**

Dipartimento di Matematica

Corso di Laurea in Informatica


# Sviluppo di un'applicazione Android per un sistema peer-to-peer di apprendimento di una lingua straniera

*Tesi di laurea triennale*


*Relatore*

Prof. Tullio Vardanega

*Laureando*

Luca De Franceschi



Anno Accademico 2014-2015

# Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa trecento ore, dal laureando Luca De Franceschi presso l'azienda CoffeeStrap inc. L'obiettivo di tale attività di stage era rivolta alla realizzazione di un'applicazione Android contestualmente funzionale alla logica di business dell'azienda ospitante. Tale applicazione è stata sviluppata nativamente utilizzando l'SDK ufficiale messo a disposizione dal sistema Android e si presenta in linea con l'applicazione web, integrando al suo interno le API messe a disposizione dal backend e mantenendo una stretta sincronia tramite il sistema di notifiche push.

*"Life is really simple, but we insist on making it complicated"*

— Confucius

# Ringraziamenti

*Innanzitutto, vorrei esprimere la mia gratitudine al Prof. Tullio Vardanega, relatore della mia tesi, per l'aiuto e il sostegno fornitomi durante la stesura del lavoro e nel corso dell'attività di stage.*

*Desidero ringraziare con affetto i miei genitori per il sostegno, il grande aiuto e per essermi stati sempre vicini in ogni momento durante gli anni di studio, soprattutto nei momenti di difficoltà.*

*Ho desiderio di ringraziare poi i compagni di corso con cui sono entrato in contatto, senza i quali non avrei nemmeno lontanamente potuto raggiungere questo obiettivo. In particolare ringrazio il mio team, gli* `SteakHolders`*, per la bellissima esperienza passata insieme e i duri mesi di lavoro che mi hanno permesso di crescere moltissimo.*

*Un ringraziamento speciale va ad Alessandro, Milo e Mahesh, fondatori di CoffeeStrap, che hanno creduto in me e mi hanno dato la possibilità di esprimere le mie capacità in questa bellissima esperienza di stage.*

*Infine vorrei ringraziare tutti i miei amici, che mi hanno sostenuto in questo percorso di studio e con i quali ho passato degli splendidi momenti.*

*Padova, Feb 2015* Luca De Franceschi

# Indice

# Elenco delle figure



# Elenco delle tabelle

# Capitolo 1

# Il contesto aziendale

## 1.1 L'azienda CoffeeStrap inc.

CoffeeStrap inc. è un'azienda start-up in ambito web/mobile nata nei primi mesi del 2013 dalla collaborazione di Mahesh Casiraghi e Alessandro Maccagnan, che attualmente ricoprono al suo interno rispettivamente i ruoli di CEO e CTO. Dal mese di Aprile del 2014 inoltre si è aggiunto al team un ulteriore componente, Milo Ertola, che attualmente ricopre il ruolo di CPO e si occupa dello sviluppo web.


**figura 1.1:** Logo di CoffeeStrap inc.

La start-up nasce con sede operativa a San Francisco, in cui ha trascorso il suo primo anno di sviluppo, per poi spostarsi all'inizio del 2014 nel cuore Amsterdam, partecipando al programma dell'acceleratore Rockstart[1], che fornisce attualmente supporto diretto all'azienda, mettendo a disposizione, oltre alle postazioni di lavoro (è infatti anche incubatore), un ambiente fresco e innovativo, circondato da esperti del settore e a stretto e costante contatto con il mondo esterno e con la rete degli investitori. Rockstart possiede al suo interno inoltre un gruppo di **mentori**, che mettono a disposizione la loro esperienza e le loro capacità, fornendo assistenza diretta in diversi ambiti. Questo tipo di figura è molto importanti in un ambiente di questo tipo, in quanto le aziende, essendo alle origini, hanno bisogno di essere indirizzate nella giusta direzione per poter sviluppare il proprio prodotto in modo efficace e ricevere dunque finanziamenti.

La sede operativa contestualmente al mio stage è stata appunto quest'ultima, nella quale ho potuto conoscere il mondo delle start-up ed entrare in contatto con una comunità di sviluppatori, esperti di business e mentori che hanno contribuito alla mia crescita sia professionale che personale.

[1] http://www.rockstart.com/accelerator/

**figura 1.2:** Logo di Rockstart Accelerator

### 1.1.1 Il contesto della start-up

L'azienda ospitante è una particolare realtà aziendale, l'impresa è nella sua fase iniziale e, come la maggior parte delle start-up, parte con un'idea di riferimento, cercando di rendere profittevole quest'ultima tramite veloci iterazioni sul prodotto. L'incubatore all'interno del quale lavoravo incapsula questo mondo in maniera molto significativa. Sono entrato in un ambiente molto personalizzante, nel quale si è a stretto contatto con il team (essendo quest'ultimo inoltre di piccole dimensioni) e le altre aziende, con le quali si è creata un'implicita rete collaborativa, dalla quale è possibile imparare molto e confrontarsi quotidianamente. Tale collaborazione fornisce valore al prodotto e all'azienda, in quanto permette, oltre ad un importante feedback sulle iterazioni, anche una quotidiana attività di brainstorming, che permette di visualizzare i problemi sotto punti di vista differenti e ad arrivare così ad una migliore soluzione.

## 1.2 Il dominio applicativo

L'idea di CoffeeStrap è quella di fornire un sistema peer-to-peer per l'apprendimento di una lingua straniera tramite interazioni con altre persone che parlano quella determinata lingua. L'intento è quello di creare una comunità di utenti che vogliano imparare una lingua e siano allo stesso tempo disposti a fornire supporto ad altri utenti con le lingue cui sono già fluenti. L'interazione può avvenire tramite **comunicazione testuale**, in cui viene messa a disposizione una vera e propria *chat*, oppure tramite **comunicazione audio/video**, in cui gli utenti parlano tra di loro nella lingua concordata. Gli utenti hanno la possibilità di iscriversi tramite email o tramite facebook. Il profilo utente è caratterizzato dai seguenti campi:

- Nome;
- Cognome;
- Età (opzionale);
- Immagine profilo;
- Città di provenienza;
- Lingue conosciute;
- Lingue da imparare.

Una volta completato il profilo l'utente inizia a parlare una lingua passando attraverso un flusso:

- L'utente $X$ seleziona la lingua $Y$ che vuole praticare;

- Il sistema cerca all'interno della comunità un sottoinsieme $Z$ di persone che parlano la lingua $Y$ (come lingua madre o seconda lingua);
- Il sistema notifica quest'ultimo, informando gli utenti che l'utente $X$ vuole praticare la lingua $Y$;
- Gli utenti del sottoinsieme $Z$ decidono se accettare o meno la richiesta dell'utente $X$ di praticare la lingua $Y$;
- Se l'utente accetta la richiesta allora viene creato un **match** tra quest'ultimo e l'utente $X$, ed essi iniziano un'interazione.

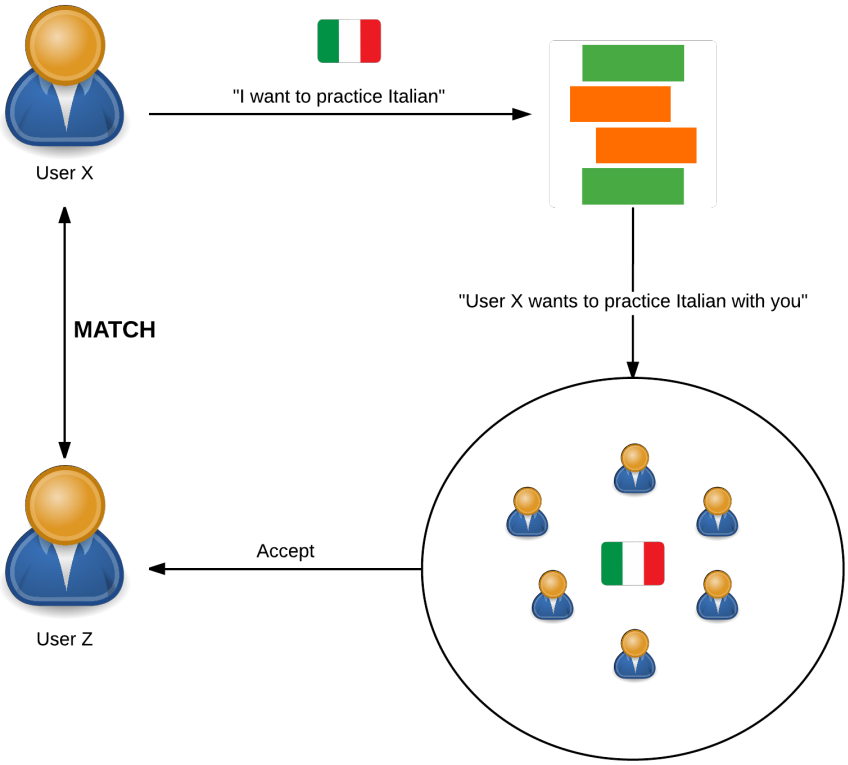



**figura 1.3:** Flusso utente

Gli utenti vengono notificati tramite **notifica email** e, in seguito allo sviluppo dell'applicazione, **notifica mobile**.

## 1.3 Metodologie e strumenti di sviluppo

CoffeeStrap adotta la metodologia di sviluppo **agile**. Il particolare contesto entro il quale si trova fa sì che i requisiti cambiano molto spesso e porta dunque quest'ultima a dover iterare molto velocemente sul proprio prodotto, che si trova ancora nella sua fase primordiale e necessita di modifiche e miglioramenti costanti. Proprio a fronte di

queste necessità è importante mantenere la massima flessibilità possibile nello sviluppo, motivo per cui questo modello di ciclo di vita è in questo momento il più conveniente in termini di efficienza ed efficacia. In particolare la metodologia adottata è quella **agile** **scrum**.

Il tipo di iterazione utilizzata è su base settimanale. Una volta a settimana, normalmente il Mercoledì, avviene lo **scrum meeting**, una riunione della durata di circa due ore nella quale si fissano **stories** da risolvere nello **sprint** corrente. Una story è un concetto molto simile a quello di **ticket**, ed è composta dai seguenti campi:

- **Titolo**;
- **Tipologia**, che può assumere i seguenti valori:
  - **Feature**, quando c'è da aggiungere un "pezzo" al sistema;
  - **Bug**, quando c'è da risolvere una problematica all'interno del sistema;
  - **Chore**, quando c'è da eseguire *refactoring* e manutenzione del codice;
  - **Release**, sono marcatori di *milestone* che tracciano il progresso del team rispetto agli obiettivi.
- **Descrizione**, che fornisce una dettagliata sintesi della story.
- Una o più **etichette**, che forniscono un filtro alle stories;
- **Complessità**, che è un indice soggettivo di quante risorse sono necessarie per il completamento della story. Questo valore segue la sequenza di Fibonacci e può assumere i valori 1, 2, 5, 8, 13. Se una story ha una complessità superiore a 13 essa dev'essere necessariamente spezzata in due o più sotto-stories. Solamente alle features può essere assegnata complessità, in quanto sono le uniche stories che realmente forniscono un valore al business;
- **Richiedente**, che dev'essere unico;
- Uno o più **responsabili** (*owners*), che possono coincidere con il richiedente e sono coloro che devono occuparsi direttamente della story;

Opzionalmente inoltre è possibile aggiungere alla story dei tasks, in modo da dividere ulteriormente il lavoro. Ciononostante su di essi non vi è alcun controllo, in quanto il loro utilizzo è a discrezione dei responsabili.

Ciascuna story è inserita all'interno di una **board**. Ci sono tre tipologie di board:

- **Current**, contenente le stories attive nello sprint corrente;
- **Backlog**, contenente la "coda" delle stories da attivare;
- **Icebox**, contente stories in attesa di approvazione e attivazione. Le stories possono rimanere su questa board per un tempo indeterminato, finché vengono spostate sul backlog o rimosse definitivamente.

| **Titolo** | Login/registrazione con Facebook |
|---|---|
| **Tipologia** | Feature |
| **Descrizione** | È necessario interfacciarsi con le API di facebook per Android per poter implementare il sistema di login/registrazione tramite facebook. Una volta ottenuto il token di accesso l'applicazione deve effettuare una chiamata API a CoffeeStrap fornendo quest'ultimo e memorizzando il cookie restituito dalla chiamata. |
| **Etichette** | `Mobile` |
| **Complessità** | 8 |
| **Richiedente** | Alessandro Maccagnan |
| **Responsabili** | Luca De Franceschi |

**tabella 1.1:** Un esempio di story

A ciascun membro del team vengono assegnate una o più stories, la cui scadenza è lo scrum meeting successivo. A quest'ultimo viene richiesto inoltre di assegnare la complessità a quella story, la quale dovrà essere calcolata secondo criteri derivati dalla propria esperienza di sviluppo. Ogni story dev'essere analizzata, progettata, sviluppata e infine testata. La chiusura della story viene effettuata solamente al seguito di una verifica e validazione da parte del responsabile e del richiedente o, se il richiedente coincide con il responsabile, da un altro membro del team. In generale i responsabili non possono approvare le proprie stories.

Una volta creata la story deve passare attraverso diverse fasi, per questo si parla di **story workflow**:

1. La story viene **creata** e **definita** (*define and design*);
2. I responsabili assegnano loro una **stima di complessità**, che va generalmente discussa all'interno del team (*estimate*);
3. La story viene **attivata**: da questo momento i responsabili possono procedere allo sviluppo (*start*);
4. La story una volta implementata viene **testata** (*test*);
5. Superata la fase di testing la story viene **ultimata** (*finish*);
6. A questo punto i responsabili "pushano" il codice sull'ambiente di **staging** (*deliver*);
7. La story viene poi **accettata** dal richiedente o comunque da una persona esterna al gruppo di responsabili e viene effettuato il pushing sull'ambiente di **production** (*accept*). Se la story non viene accettata si itera dal punto 4;
8. Infine, se la story è stata accettata avviene il suo **rilascio**, che coincide con la terminazione del suo ciclo di vita (*release*).

Alla fine dello sprint viene tracciata la somma delle complessità di tutte le stories assegnate a tutti i membri e che sono state chiuse nello sprint corrente. Tale somma è detta **velocity**, ed è un indice di efficienza di sviluppo dello sprint, che va confrontato con gli sprint precedenti.

Oltre allo scrum meeting viene instanziato quotidianamente lo **stand-up meeting**, detto anche **daily scrum**, nel quale prima dell'inizio della giornata lavorativa si

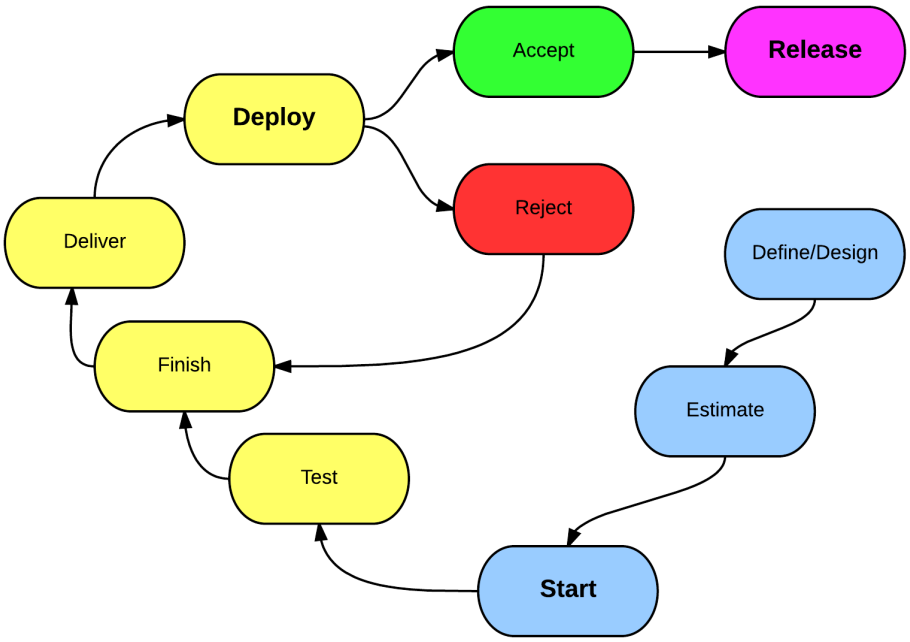



**figura 1.4:** Il flusso di una story

effettua una breve riunione con il team, in cui ciascuno espone quanto fatto nel giorno precedente e ciò che ha in programma di fare nel giorno corrente. Quest'attività è di fondamentale importanza per **monitorare** il progresso dello sprint e sollevare eventuali difficoltà riscontrate nello svolgimento della story.

## 1.4 Tecnologie

### 1.4.1 Back-end

Il back-end è composto da un server scritto tramite il framework Node.js. Questa scelta è dovuta alla grande potenza e scalabilità di quest'ultimo, oltre al fatto che si tratta di un linguaggio di programmazione all'avanguardia la cui comunità è in forte espansione. Inoltre tramite **npm**, gestore di pacchetti per Node.js, è possibile attingere ad una vasta quantità di librerie.

Il server fornisce tutta una serie completa di **API RESTful** accessibili dall'esterno, alcune delle quali richiedono autenticazione. Ciò permette a tutti i client che si interfacciano con esso, in particolare l'applicazione web e l'applicazione mobile, di poter comunicare con esso in modo semplice e lineare, dovendo riferirsi solamente alle specifiche di queste ultime ed utilizzando una libreria per la gestione delle chiamate API.

Parallelamente al server è stato instanziato un servizio di **pushing** fornito da **firebase**, un **Cloud Service Provider** utilizzato per il sistema di notifiche e per fornire un punto d'accesso centralizzato dal quale i diversi componenti (intesi come componenti software) possono comunicare. Firebase fornisce infatti un **database in**

**figura 1.5:** Logo di Node.js

**tempo reale**, con il quale è possibile interagire e "*mettersi in ascolto*" degli eventi su di esso. Esso fornisce un'API pubblica per diversi stack tecnologici e piattaforme.

**figura 1.6:** Logo di Firebase

### 1.4.2 Front-end

Il front-end è composto da un'applicazione web scritta con **Angular.js** e **HTML5**, utilizzando il framework CSS **Bootstrap**. Tramite Angular.js è possibile interfacciarsi alle API fornite dal back-end in maniera molto efficiente e popolare dunque le pagine in base alle risposte provenienti da esso. Inoltre, analogamente a npm, per lo sviluppo di applicazioni con Angular.js è disponibile **bower**, gestore di pacchetti per quest'ultimo.

**figura 1.7:** Logo di Angular.js

### 1.4.3 Database

CoffeeStrap utilizza **MongoDB** come database documentale di riferimento. Questa scelta è dovuta alla grande flessibilità offerta dai database di questo tipo, oltre al fatto che negli ultimi mesi è divenuto molto popolare nella comunità di sviluppatori e presenta prestazioni notevoli.

Dal momento che i requisiti cambiano molto rapidamente è necessario avvalersi di strumenti che siano il più flessibili possibili ai cambiamenti, e MongoDB risponde perfettamente a questo tipo di esigenza. Inoltre MongoDB presenta un facile approccio al suo utilizzo sia per l'utilizzo da parte dello sviluppatore che in termini di scalabilità.

### 1.4.4 Ambienti di sviluppo

All'interno dell'azienda si delineano principalmente due ambienti di sviluppo:


**figura 1.8:** Logo di MongoDB

- **Production**, all'interno del quale risiedono le versioni ufficiali del codice esposte pubblicamente agli utenti;
- **Staging**, un ambiente riservato esclusivamente al team e che rappresenta una sorta di "cantiere", nel quale risiede una versione del codice ancora da verificare, che può quindi essere testata liberamente e non crea danni verso l'esterno.

Ciascun ambiente è caratterizzato dalla propria configurazione, possiede il proprio server e il proprio database ed è agnostico rispetto all'altro.

Occasionalmente è inoltre uso comune instanziare un ambiente ausiliario di sviluppo, per circostanze particolari che necessitano di una configurazione diversa. Solitamente sono ambienti "*usa e getta*", che nascono da esigenze particolare e vengono rimossi alla fine del loro utilizzo.

### 1.4.5 Versionamento

Come strumento di controllo versione del codice sorgente l'azienda utilizza **Git**, utilizzando **Bitbucket** come servizio di hosting. I motivi che hanno spinto all'utilizzo di Git in luogo di altri sistemi di versionamento sono i seguenti:

- **Disponibilità**: ciascun sviluppatore ha una copia locale dell'intero repository ed effettua *commit* in locale, potendo dunque lavorare anche in assenza di connessione;
- **Semplicità**: oltre ad essere veloce Git permette con facilità la creazione di *branch* e successivi *merge*. L'utilizzo di questi ultimi è fondamentale per poter soddisfare il punto seguente:
- **Utilizzo di Gitflow**: questo **workflow** definisce uno stretto modello di *branching* che ruota intorno al progetto. Ciò non aggiunge nuovi concetti o strumenti a Git, ma piuttosto assegna ruoli specifici ai diversi rami. Un approfondimento di questa metodologia è raccolta nell'appendice A.


**figura 1.9:** Logo di Git

## 1.5 Clientela di riferimento

CoffeeStrap non ha ancora acquisito finanziamenti da parte di investitori e non presenta associazioni con altre aziende o entità, motivo per cui essi devono rispondere unicamente agli **utenti finali** del prodotto. Questi ultimi sono persone reali provenienti da ogni parte del mondo, che mettono a disposizione le proprie competenze linguistiche e vogliono allo stesso tempo imparare una lingua straniera entrando in contatto con altri utenti. L'intera struttura di CoffeeStrap si basa su questa **rete sociale**, composta da un insieme eterogeneo di persone che espongono il proprio profilo all'esterno. È possibile immaginare CoffeeStrap come un vero e proprio *social network* il cui scopo è l'apprendimento linguistico.


**figura 1.10:** Distribuzione degli utenti di CoffeeStrap nel mondo aggiornata a Dicembre 2014

# Capitolo 2

# Il progetto all'interno dell'azienda

## 2.1 Presentazione del progetto

In questa sezione fornirò un'introduzione del progetto in questione.

### 2.1.1 Motivazioni

In questa sezione descriverò le motivazioni che hanno portato CoffeeStrap ad ospitare uno stage all'interno dell'azienda, quali vantaggi esso avrebbe portato e quale valore aggiunto si avrebbe avuto alla fine.

### 2.1.2 Aspettative

In questa sezione descriverò cosa l'azienda CoffeeStrap si aspettava da me e dal prodotto all'inizio dello stage.

### 2.1.3 Obiettivi di formazione

In questa sezione descriverò la strategia assunta da CoffeeStrap all'inizio dello stage legato al personale, ovvero al voler aggiungere un componente, un ruolo all'interno del team di sviluppo, legato anche al fatto che essi avevano già lavorato con me in precedenza nel progetto di Ingegneria del Software e quindi erano consci di quali fossero le skill da me acquisite.

## 2.2 Vincoli

### 2.2.1 Vincoli tecnologici

In questa sezione descriverò quali sono stati i vincoli tecnologici imposti per la realizzazione del progetto, come ad esempio gli ambienti di sviluppo, i framework e le librerie utilizzate. In particolare approfondirò le motivazioni che hanno portato alla scelta di sviluppare nativamente in Android piuttosto che utilizzare framework di sviluppo quali Titanium o Ionic, che erano stati inizialmente considerati.

### 2.2.2 Vincoli metodologici

In questa sezione descriverò le metodologie di sviluppo imposte dall'azienda per lo sviluppo del progetto.

### 2.2.3 Vincoli temporali

In questa sezione descriverò quali sono stati i vincoli temporali per la realizzazione degli obiettivi preposti.

# Capitolo 3

# Il progetto di stage

## 3.1 Pianificazione del lavoro

In questa sezione descriverò come il progetto è stato suddiviso nell'arco delle otto settimane e quali erano gli obiettivi alla fine di ciascun periodo.

## 3.2 Norme, procedure e strumenti

In questa sezione descriverò tutte le norme e le procedure con il quale ho organizzato il mio lavoro, quali processi ho attuato con le loro motivazioni e di quali strumenti mi sono avvalso per l'attuazione di tali processi.

## 3.3 Periodo di formazione

### 3.3.1 Apprendimento della tecnologia

In questa sezione descriverò come ho appreso la tecnologia di riferimento e attraverso quali processi, approfondendo nel dettaglio alcuni concetti indispensabili e di rilievo. Descriverò la differenza tra le mie conoscenze alla fine di questo periodo e all'inizio, fornendo evidenza di apprendimento.

### 3.3.2 Integrazione con le tecnologie aziendali

In questa sezione descriverò come, dopo uno studio della tecnologia legata al progetto di stage abbia integrato quest'ultima con le tecnologie già esistenti utilizzate all'interno dell'azienda.

## 3.4 Progettazione

### 3.4.1 Progettazione del flusso utente

In questa sezione descriverò la progettazione del flusso utente all'interno dell'applicazione Android, avvalendomi di diagrammi di attività e di flusso.

### 3.4.2 Progettazione del layout

In questa sezione descriverò la progettazione del layout delle diverse schermate dell'applicazione, giustificando le varie scelte secondo criteri di accessibilità e usabilità.

### 3.4.3 Progettazione architetturale

In questa sezione descriverò la progettazione dell'architettura software dell'applicazione, andando ad evidenziare la suddivisione in classi e pacchetti, le interazioni di quest'ultimi, le librerie esterne e i design pattern utilizzati; il tutto in relazione con le best-practise indicate dalla comunitò di sviluppatori Android.

## 3.5 Sviluppo e codifica

In questa sezione descriverò come ho realizzato il codice dell'applicazione aderendo alla progettazione attuata in precedenza.

## 3.6 Rifinitura e testing

In questa sezione descriverò come a fronte dello sviluppo e codifica dell'applicazione ho proceduto alla sua verifica, istanziando un sistema di bug tracking e testando l'applicazione con utenti reali.

## 3.7 Validazione del prodotto e rilascio

In questa sezione infine descriverò come a fronte dell'attività di rifinitura è stata istanziata un'attività di validazione mirata a verificare che il prodotto corrispondesse alle aspettative e soddisfasse gli obiettivi preposti. Inoltre descriverò l'attività di rilascio e di pubblicazione di una versione sul play store di Google.

# Capitolo 4

# Valutazioni retrospettive

## 4.1 Bilancio sui risultati

In questa sezione darò evidenza di aver soddisfatto tutti gli obiettivi di stage, fornendo un consuntivo di quanto realizzato, confrontato con il preventivo di inizio stage.

## 4.2 Bilancio formativo

In questa sezione confronterò la mia formazione all'inizio dello stage con quella ottenuta alla fine, evidenziando quanto appreso e il tipo di maturazione ottenuta sul piano tecnico e metodologico.

## 4.3 Distanza tra contesto lavorativo e mondo accademico

In questa sezione fornirò un GAP di conoscenze, evidenziando gli elementi nulli sul quale il corso di laurea non è riuscito a formarmi. Fornirò dunque una critica costruttiva nei confronti del mondo accademico, confrontando il mio bagaglio in ingresso con quello in uscita e presentando dei suggerimenti in ottica di miglioramento del corso di laurea.

# Appendice A

# Gitflow

Gitflow è un modello di sviluppo basato su Git, ideato da Vincent Driessen e presentato nel 2010 in un suo celebre post.[1]. Sebbene sia pittosto complicato rispetto ad altri modelli, questo framework fornisce un robusto strumento per la gestione di progetti di grandi dimensioni.

Gitflow assegna ruoli specifici ai diversi branch, specificando quando e come essi debbano interagire tra di loro. Essendo basato su Git gli sviluppatori lavorano in locale e periodicamente effettuano *push* sul repository remoto centrale.

## A.1 I branch principali

Nel repository principale si distinguono due rami principali:

- **master**, il ramo principale all'interno del quale risiede codice verificato e pronto per essere spedito in *production* e in particolare tutti i suoi nodi corrispondono a **release** del prodotto, ovvero ad una versione stabile, che può essere marcata o meno con un *tag*;

- **develop**, il ramo all'interno del quale avviene lo sviluppo vero e proprio del prodotto e dal quale il codice può essere rilasciato, effettuando dunque un *merge* con il ramo master

## A.2 I branch di supporto

Al livello inferiore dei branch principali troviamo i branch di supporto, che assistono gli sviluppatori nelle operazioni quotidiane di sviluppo. Questi branch, a differenza dei due principali, hanno un tempo di vita limitato e vengono uniti nei rami principali o semplicemente scartati. I tre branch di supporto sono:

- **Feature**, che viene creato dal ramo `develop` ed unito solamente nel medesimo; la loro utilità, come dice la parola stessa, è quella di creare un ramo di sviluppo per un nuovo componente del sistema. Essi normalmente vanno tenuti in locale e non devono essere *pushati* sul repository remoto;

[1] http://nvie.com/posts/a-successful-git-branching-model/

- **Release**, che viene creato dal ramo `develop` ed unito sia nel medesimo che successivamente su `master`. Questo branch identifica le procedure che vengono istanziate all'atto della release. Quando uno sviluppatore decide di effettuare una release del prodotto apre innanzitutto un brach da `develop` chiamato `release-*`, esegue tutte le operazioni di *pre-release*, effettua il merge su `develop` ed infine il merge su `master`, marcando il nodo di merge eventualmente con un *tag*;
- **Hotfix**, che viene creato dal ramo `master` ed unito prima sia nel medesimo che successivamente su `develop`. Questi rami derivano dall'immediata necessità di risolvere un *bug* inserito all'interno del ramo master, senza dover effettuare una nuova release.

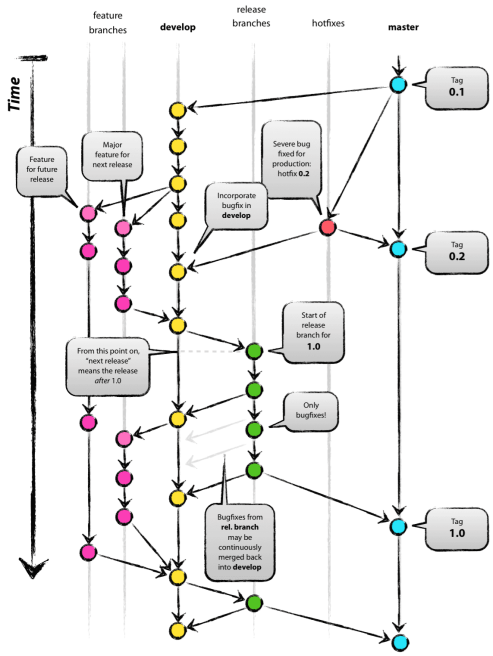



**figura A.1:** Modello Gitflow

# Glossario

**acceleratore** Nell'ambito della startup l'acceleratore indica un'associazione che fornisce sostegno e assistenza all'azienda nella sua fase primordiale, "accelerando" il suo processo di sviluppo mettendo quest'ultima nell'ambiente giusto e a stretto contatto con il mondo esterno e la rete degli investitori. Normalmente queste associazioni organizzano un vero e proprio programma di accelerazione, fornendo un investimento iniziale all'azienda. In cambio solitamente l'acceleratore detiene una quota societaria.. 1, 19

**agile** Nell'ingegneria del software si riferisce ad una metodologia di sviluppo del software contrapposto ai modelli tradizionali (ad esempio *waterfall*), in cui si pone l'attenzione sul consegnare al cliente il prodotto in tempi brevi e con iterazioni veloci. Tale metodologia è rappresentata dal *Manifesto Agile*.[2]. 3, 19

**Angular.js** Si tratta di un framework per lo sviluppo web mantenuto dalla Google creato nel 2009. Favorisce un approccio di tipo dichiarativo allo sviluppo *client-side*, migliore per la creazione di interfacce utente, laddove l'approccio imperativo si presta maggiormente per la realizzazione della logica applicativa. Esso si ispira fortemente al design pattern MVC riducendo considerevolmente il codice necessario alla realizzazione di applicazioni HTML/javascript.. 7, 19

**API** Indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma. La finalità è ottenere un'astrazione, di solito tra l'hardware e il programmatore o tra software a basso e ad alto livello semplificando così il lavoro di programmazione.. 6, 19

**Bootstrap** È una raccolta di strumenti liberi per la creazione di siti e applicazioni per il web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia, che per le varie componenti dell'interfaccia, come moduli, bottoni e navigazione, e altri componenti dell'interfaccia, così come alcune estensioni opzionali di JavaScript.. 7, 19

**brainstorming** Tecnica eseguibile in gruppo volta alla raccolta di idee per la soluzione di un problema.. 2, 19

**CEO** Traducibile in italiano come *amministratore delegato*, nel mondo delle startup è un componente del team che si occupa normalmente del lato *business* dell'impresa, è colui che fornisce un'interfaccia tra l'azienda e il mondo esterno.. 1, 19

[2] http://agilemanifesto.org/

**Cloud Service Provider** Si tratta di una struttura o un'organizzazione che fornisce servizi cloud ai clienti, a fronte di un contratto di fornitura.. 6, 20

**commit** Nell'ambito di un repository indica l'operazione con cui si genera una versione del proprio lavoro, aggiungendo un nodo al ramo di sviluppo.. 8, 20

**CPO** È un responsabile di alto livello all'interno di una startup che si occupa della gestione e coordinazione dei processi aziendali.. 1, 20

**CTO** È il responsabile del comparto tecnico all'interno di una startup che si occupa della valutazione e selezione delle tecnologie che possono essere applicate al prodotto o ai servizi che l'azienda produce.. 1, 20

**Git** È un sistema software di controllo di versione distribuito, creato da Linus Torvalds nel 2005.. 8, 20

**HTML5** È un linguaggio di markup per la strutturazione delle pagine web, pubblicato come W3C Reccomendation nell'Ottobre del 2014. Le novità introdotte dall'HTML5 rispetto all'HTML4 sono finalizzate soprattutto a migliorare il disaccoppiamento fra struttura, definita dal markup, caratteristiche di resa, definite dalle direttive di stile, e contenuti di una pagina web, definiti dal testo vero e proprio. Inoltre l'HTML5 prevede il supporto per la memorizzazione locale di grosse quantità di dati scaricati dal web browser, per consentire l'utilizzo di applicazioni basate su web anche in assenza di collegamento a Internet.. 7, 20

**incubatore** Il concetto di incubatore è molto simile a quello di *acceleratore*, indicando più precisamente il "luogo fisico" all'interno del quale risiede e lavora la startup.. 1, 20

**merge** Nell'ambito di un repository indica la procedura tramite la quale un *branch* viene unito in un altro, generalente nel branch che lo ha creato.. 8, 20

**MongoDB** È un database non relazionale (NoSQL), orientato ai documenti. Classificato come un database di tipo NoSQL, MongoDB si allontana dalla struttura tradizionale basata su tabelle dei database relazionali in favore di documenti in stile JSON con schema dinamico (MongoDB chiama il formato BSON), rendendo l'integrazione di dati di alcuni tipi di applicazioni più facile e veloce.. 7, 20

**Node.js** È un framework *event-driven* di utilizzo *server-side* di Javascript. 6, 20

**peer-to-peer** È un'espressione che indica un'architettura logica di rete informatica in cui i nodi non sono gerarchizzati unicamente sotto forma di client o server fissi (clienti e serventi), ma sotto forma di nodi equivalenti o paritari (in inglese peer) che possono cioè fungere sia da cliente che da servente verso gli altri nodi terminali (host) della rete. Nell'ambito di CoffeeStrap indica che l'utente svolge sia il ruolo di "insegnante" che quello di "studente" per l'apprendimento linguistico.. 2, 20

**REST** Riferisce ad un insieme di principi di architetture di rete, i quali delineano come le risorse sono definite e indirizzate. Il termine è spesso usato nel senso di descrivere ogni semplice interfaccia che trasmette dati su HTTP.. 6, 20

**scrum** È un framework di sviluppo agile del software iterativo ed incrementale in cui vengono enfatizzati tutti gli aspetti di progetto legati a contesti in cui è difficile pianificare in anticipo e in cui generalmente i requisiti cambiano molto spesso. Vengono istanziati meccanismi di controllo empirico, in cui al principio di *"command-and-control"* viene contrapposta una tecnica di gestione basata su rapidi **cicli di feedback**.. 4, 21

**sprint** Nella metodologia scrum viene inteso come il lasso di tempo che intercorre tra uno scrum meeting e il successivo. In particolare è la misura della dimensione di un'iterazione sul prodotto.. 4, 21

**start-up** Identifica la fase iniziale per l'avvio di una nuova impresa, cioè quel periodo nel quale un'organizzazione cerca di rendere profittevole un'idea attraverso processi ripetibili e scalabili. Inizialmente il termine veniva usato unicamente per indicare la fase di avvio di aziende nel settore internet o tecnologie dell'informazione.. 1

# Bibliografia

*Manifesto Agile*. URL: http://agilemanifesto.org/.

*Scrum Methodology*. URL: http://scrummethodology.com/.

## Riferimenti bibliografici

## Siti Web consultati

*Manifesto Agile*. URL: http://agilemanifesto.org/.