UNIVERSITÀ DEGLI STUDI DI SALERNO

Facoltà di Scienze Matematiche Fisiche e Naturali

Tesi di Laurea di I livello in

INFORMATICA

ABSTRACT

Miniaturizzazione di applicazioni Java Desktop in App Mobile: gestione della tabelle

Relatore

Prof. Michele Risi

Candidato

Egidio Giacoia

Matr. 051210/2376

Anno accademico 2016/2017

Nell'ultimo decennio con il progredire della tecnologia il mercato degli **smartphone** continua ad espandersi ed evolversi sempre più rapidamente, diventando uno strumento fondamentale nella vita quotidiana delle persone. Oltre alla vendita dei dispositivi mobile cresce, di conseguenza, anche il mercato delle applicazioni messe a disposizione per un vasto bacino di utenti.

Le aziende per stare al passo con i tempi stanno focalizzando il proprio interesse su questo tipo di tecnologie. Tuttavia questo implica un notevole sforzo, in termini di risorse e tempo, nel riscrivere completamente da zero i sistemi pensati per essere eseguiti su **ambiente desktop,** trasformandoli in sistemi per essere eseguiti su **ambiente mobile**.

L'idea di base dell'attività di tesi è nata proprio da quest'ultimo concetto, ovvero di **miniaturizzazione** di un software, i.e., trasformare un'applicazione desktop in un'applicazione utilizzabile sullo smartphone. Tale attività è legata alla **portabilità** (dall'inglese *porting*), che consiste nell'adattare il software in modo tale che un programma applicativo possa essere eseguito in un ambiente computazionale diverso da quello per cui era stato progettato inizialmente. Le cause che possono portare ad effettuare richieste di operazioni di porting sono dovute, per esempio, a delle diversità di utilizzo delle CPU, delle API, dei sistemi operativi, dell'hardware o dall'incompatibilità dell'implementazione del linguaggio di programmazione sull'ambiente target.

L'intero processo può risultare un'attività complessa e costosa per cui bisogna considerare vari aspetti:

- **Analizzare il dominio applicativo.**
  Si deve considerare il divario tecnologico tra l'ambiente di origine e l'ambiente di destinazione, il tipo di componente software da "portare" e gli strumenti con cui esso è stato costruito.
- **Velocizzare il Time To Market.**
  Gli investitori e sviluppatori per lanciare la propria applicazione nel mercato devono stabilire la strategia e il modello di business da adottare, la scelta dei collaboratori per lo sviluppo delle applicazioni, i budget necessari e gli obiettivi di guadagno.
- **Scegliere un modello di sviluppo dell'app mobile.**
  Lavorando in ambiente nativo si riescono ad avere applicazioni performanti e pieno accesso alle funzionalità del device, ma ciò richiede un'ottima conoscenza del linguaggio di programmazione dell'ambiente e, soprattutto, costi e tempi di sviluppo notevoli (**app nativa**);

Se si sceglie di sviluppare l'applicazione utilizzando tecnologie Web e framework si riducono costi e tempi ed inoltre si ottiene un'applicazione cross-platform, ma le prestazioni e l'accesso alle funzionalità del dispositivo sono ridotte considerevolmente (**web app**).

Una soluzione a tali problematiche è il processo di miniaturizzazione e le tecnologie sui cui si basa, che hanno permesso di realizzare un meccanismo semi-automatico il quale trasforma un'applicazione desktop Java eseguibile solo sul computer, in un'applicazione eseguibile sul sistema operativo mobile Android, attraverso l'uso del framework PhoneGap. Tutto ciò è stato realizzato creando un wrapper per l'applicazione Java di partenza in modo tale da renderla compatibile con il nuovo ambiente in cui sarà eseguita senza modificarne la logica di business, utilizzando una **libreria di supporto** e un **middleware di comunicazione** per collegare l'interfaccia utente miniaturizzata con tutte le funzioni presenti all'interno del codice considerato. Tutto ciò ha portato a definire un'**applicazione** tecnicamente **ibrida**, con alcune peculiarità:

- miglioramento delle performance in quanto la logica è nativa;
- minimizzazione dello sviluppo in quanto esso parte da una applicazione già esistente.

L'obbiettivo centrale del lavoro di tesi è stato quello di ampliare il processo di miniaturizzazione: il sistema definito permette di sviluppare nuovi componenti dell'interfaccia grafica utente gestibili in maniera automatica. L'attività svolta consiste nell'aver esteso la libreria di supporto con una nuova componente per la **gestione di tabelle e delle sorgenti di dati**: è stato realizzato un algoritmo ricorsivo che permette di effettuare il parsing di ogni componente della tabella (intestazione, righe e celle) presente nell'applicazione Java; inoltre sono state definite delle direttive per delineare un'interfaccia user-friendly ed implementata la logica d'interazione tra l'utente e la tabella sfruttando il plugin di comunicazione, con il fine di migliorare l'user-experience e aumentare la compatibilità tra l'applicazione desktop e l'applicazione mobile. Lo sviluppo di tale componente ha richiesto un notevole sforzo di comprensione su come avvenisse l'interazione con l'utente e sul funzionamento e gestione dei dati in quanto la componente Java corrispondente è un oggetto composto ed opera con numerose classi di supporto che rendono arduo il processo di porting.

La soluzione adottata per effettuare il processo di miniaturizzazione arriva a definire una conclusione efficace: utilizzando un **plug-in** e un **framework**, è stato possibile realizzare una **web app** visibile su ogni cellulare con ottimi risultati a livello di performance rispetto alle altre applicazioni, limitandone

tempi e costi di sviluppo. Gli esempi e i casi di studio analizzati hanno mostrato la validità del processo e la capacità di utilizzare la UI sul mobile.

I possibili sviluppi futuri riguardano:

- **Rendere automatico il processo di miniaturizzazione.**
  Esso richiede che alcuni passaggi siano eseguiti manualmente, ma nonostante tutto si è comunque delineato in tutte le sue parti dando indicazione di tutte le attività di pianificazione per la migrazione e delle linee guida.
- **Estensione del motore di generazione del layout.**
  Una limitazione del progetto riguarda l'impaginazione del layout in quanto l'aggiunta di componenti di grandi dimensioni nella costruzione dell'interfaccia grafica non riescono ad essere visualizzate all'interno del frame. Una possibile soluzione a questo problema è quello di estendere il frame qualora un componente copra un'aria più grande rispetto ad esso e segnalarlo all'utente.
- **Estensione della libreria pAWT.**
  Aggiungere le componenti maggiormente utilizzate nelle applicazioni Java based in modo tale da integrare nella libreria tutte le classi presenti nel pacchetto *AWT* di *Java.* In particolare si possono trattare negli sviluppi futuri la componente *Dialog* e la possibilità di poter gestire più interfacce contemporaneamente.