

The Battle for Wesnoth: Companion App - Human Computer Interaction



Alessandro Sestini

Matricola: 6226094

alessandro.sestini@stud.unifi.it

Abstract

L'uso sempre più frequente degli smartphone ha contribuito alla proliferazione di applicazioni mobili anche in ambito gaming: è infatti sempre più comune lo sviluppo di software che cercano di accompagnare e supportare l'esperienza utente di un videogioco interpretando le informazioni che si possono ricavare dai file di salvataggio. Questo progetto ha dunque come obiettivo lo studio e l'implementazione di una Companion App per sistemi Android del gioco open source "The Battle for Wesnoth", cercando di applicare i concetti tipici della Human Computer Interaction studiati a lezione. Il lavoro è stato suddiviso in varie parti: un primo processo di Needfinding, l'implementazione del software e infine dei test di usabilità.

1. Introduzione

1.1. The Battle for Wesnoth: il gioco

The Battle for Wesnoth [1] è un gioco di ruolo strategico a turni open source a tema fantastico. È un tipico gioco di strategia sulla falsa riga delle più famose saghe X-COM [4], Heroes of Might and Magic [5] o Civilization [6]: si basa sulla risoluzione di scenari in cui il giocatore comanda un party composto da più unità, ognuna con le proprie caratteristiche, mentre sono presenti altri eserciti nemici comandati dall'intelligenza artificiale che cercheranno di contrastare il giocatore. La risoluzione degli scenari avviene tipicamente con il raggiungimento di determinati obiettivi mentre l'azione va periodicamente in attesa: durante questo intervallo i giocatori possono inserire i propri comandi. Grande importanza ha dunque la gestione degli eserciti: è fondamentale essere sempre a conoscenza delle caratteristiche degli elementi in gioco, sia alleati che nemici, per poter



Fig. 1: Screenshot di una partita di gioco di The Battle for Wesnoth.

sviluppare una giusta strategia, adatta al soddisfacimento dei vari obiettivi. Lo scopo iniziale del progetto era dunque creare una applicazione Android che permettesse questo tipo di interazione anche e soprattutto in mobilità, lontano dalla propria postazione di gioco. È già presente una versione mobile del gioco, sia Android [2] che iOS [3], ma si tratta di un semplice porting di tutto il codice. In figura 1 è possibile vedere una partita avviata.

1.2. Android

In breve, Android [7] è il sistema operativo open source sviluppato da Google per sistemi mobile. Le applicazioni native per questo sistema sono sviluppate in Java e usano dei documenti XML per definire in maniera dichiarativa l'interfaccia grafica del software. Il sistema Android è risultato essere il più adatto a sviluppare un prototipo della Companion App, cercando di implementare i concetti di HCI codificati nel Material Design creato da Google stessa.

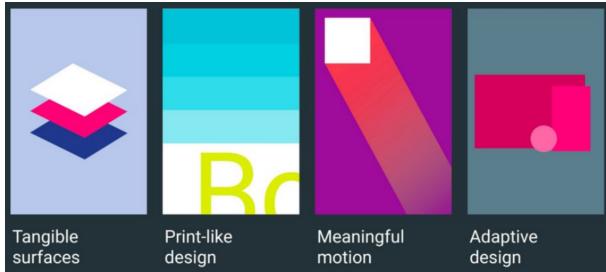


Fig. 2: I 4 principi del material design.

1.3. Material Design

Il Material Design [8] é un linguaggio di design sviluppato da Google; fa uso di layout basati su griglie e liste, animazioni responsive e specialmente dell’effetto profonditá. Si basa fortemente su delle metafore fisiche riguardanti i fogli di carta, ma estese fino ad un uso idiomatico di esse. In breve il Material Design si basa su 4 principi iniziali, visibili anche in figura 2: tangible surfaces, print-like design, meaningful motion e adaptive design. Le superfici e i contorni del ”materiale” forniscono all’utente degli indizi su come gli elementi su schermo sono disposti e interagibili: l’uso di attributi tattili familiari infatti aiuta a capire velocemente le affordance del prodotto; viene inoltre data grande importanza al movimento, utile ad attirare l’attenzione e a mantenere il flow dell’applicazione, ed a un ambiente 3D contenente luci e ombre: ogni materiale ha delle coordinate x,y e z (quest’ultima uniforme ad 1dp, dimensione tipica del material design che permette di creare degli elementi responsivi) e, tramite l’uso di luci direzionali che creano delle ombre, cerca di rappresentare la profonditá e la disposizione nell’asse z degli elementi. Questo linguaggio fa uso di elementi di design tipici e codificati, di cui vedremo i piú importanti e i piú utilizzati nella sezione riguardante l’implementazione.

2. Prima fase: Needfinding

La prima fase dello sviluppo dell’applicazione riguarda il Needfinding: tipicamente é un processo in cui si osservano dei candidati per scoprire i loro bisogni, obiettivi e valori riguardante un particolare prodotto. Da questi si costruiscono delle Personas, dei requisiti e degli scenari da cui estrarre poi una prima idea del software.

2.1. Interviste

Le interviste sono state fatte in 2 modalitá:

- È stato aperta una discussione sul forum ufficiale del gioco in cui si richiedeva di rispondere a delle domande generali per cercare di valutare i seguenti aspetti: l’interesse verso una companion app; il contesto

e le modalitá di uso di un’app del genere (assieme al gioco stesso o lontano dal PC); le informazioni e le interazioni che dovrebbe avere l’applicativo nei vari contesti. Non possiamo conoscere le caratteristiche come etá o genere di questa tipologia di candidati, che comunque comprendono sia giocatori sia sviluppatori del gioco stesso: le risposte e i consigli ricevuti sono infatti risultati molto tecnici e specifici;

- Sono state selezionate delle persone comuni di etá compresa tra i 18 e i 26 anni, di ogni genere, cercando di prendere principalmente dei videogiocatori occasionali che quindi avessero giá delle conoscenze riguardo il genere. A questi é stata proposta una sessione di gioco in cui hanno provato vari aspetti di Wesnoth, che di solito comprendevano il tutorial e uno o due scenari piú o meno avanzati di una campagna. Alla fine della sessione si richiedeva ai candidati di rispondere alle stesse domande fatte nella modalitá precedente; in questo caso le risposte ricevute sono risultate essere piú generiche, che riguardavano non tanto il gioco in sé ma l’uso di una companion app in generale.

Da queste interviste é emerso un certo interesse verso una companion app che però comprendesse solo pochi elementi, in quanto la maggior parte dei candidati avrebbe reputato ingiocabile un’eventuale versione mobile di Wesnoth. Gli intervistati si sono quindi focalizzati sugli aspetti importanti e fondamentali che potessero completare l’esperienza di gioco: la maggioranza ha ritenuto comodo un modo rapido e intuitivo di visionare le unitá sia nemiche che alleate di uno scenario attualmente in gioco, in piú la seconda tipologia degli intervistati ha fatto notare come potrebbe essere utile un modo di tenere traccia di eventuali strategie che possono venire in mente lontano dal PC, una volta che il gioco é stato salvato e chiuso.

2.2. Personas

Dalle interviste sono state dunque create 2 tipologie di Personas:

- Marco, 18-26 anni, é videogiocatore appassionato che si definisce hardcore. Frequenta l’universitá della sua cittá e passa molto tempo fuori per studiare o lavorare. Ogni giorno esce di casa la mattina, torna nel pomeriggio o la sera e si concede qualche ora di gioco; possiede uno smartphone che usa spesso sia quando é in casa che fuori, e in generale ha delle buone conoscenze riguardo l’informatica e l’uso di computer e sistemi software;
- Giulio, 20-26 anni, é un ragazzo lavoratore che passa molto tempo fuori casa. Possiede uno smartphone che

usa per la maggior parte del tempo quando é fuori e ha un laptop che usa occasionalmente per semplici compiti o per giocare quando ha del tempo libero. In generale non ha una buona conoscenza dell'informatica.

2.3. Scenari e requisiti

Le personas precedentemente create sono state quindi posizionate all'interno di possibili scenari con lo scopo di ricavare dei requisiti che l'applicazione dovrebbe soddisfare. Degli esempi di scenari individuati sono:

- Prima di andare al lavoro Giulio ha sempre voglia di farsi una sessione di gioco a Wesnoth, lasciando molto spesso una battaglia nel mezzo dell'azione. Giulio si sposta spesso tramite mezzi pubblici e gli piacerebbe controllare le proprie unitá per vedere se é possibile attuare qualche strategia: é quindi necessario che Giulio sia a conoscenza delle statistiche di tutti i personaggi in gioco. Una volta controllata la situazione, gli piacerebbe inoltre scriversi qualche promemoria per poterlo leggere quando riprenderá a giocare;
- Marco sta effettuando una campagna molto impegnativa, e molti dei suoi amici giocano a Wesnoth: Marco pensa quindi che sarebbe utile quando é con loro far vedere la propria situazione e la propria strategia precedentemente pensata per raccogliere diverse opinioni;
- Giulio sta effettuando una campagna a Wesnoth ma non si ricorda bene tutte le caratteristiche delle varie unitá che vede sullo schermo: gli piacerebbe dunque avere un sistema staccato dal computer stesso con cui si possa visionare facilmente tutte le informazioni disponibili e necessarie per completare al meglio uno scenario.

Dall'analisi di questi e altri scenari piú o meno complessi sono stati ricavati dei requisiti principali:

- L'applicazione deve permettere di caricare un determinato salvataggio;
- Gli utenti devono poter vedere lo scenario attualmente in gioco: la mappa, le unitá, gli obiettivi e le informazioni principali;
- Grande importanza hanno le informazioni relative ai personaggi: l'applicazione deve quindi mostrare tutte le piú rilevanti relative alle unitá;
- L'applicazione deve permettere inoltre di andare piú a fondo: deve poter mostare le informazioni generali di tutte le classi attualmente scoperte;

- L'applicazione deve infine permettere di tenere traccia di strategie che possono venire in mente visionando lo stato del gioco: deve quindi fornire un modo per appuntarsi qualche nota, sia in generale che riguardo ai singoli personaggi.

Sono stati inoltre definiti dei requisiti piú generali riguardo l'uso ottimale di un'applicazione mobile, ad esempio reattività, responsività e chiarezza dei contenuti.

2.4. Mock-up

Finita la fase di ricerca dei requisiti, il passo successivo é quello di farsi una prima idea dell'applicazione. A questo scopo é utile creare dei primi mock-up, in cui viene disegnato qualche sketch di come potrebbero apparire le varie schermate dell'applicazione e di come queste potrebbero interagire con l'utente per soddisfare i requisiti del capitolo precedente. In figura 3 é possibile vedere i primi disegni del software.

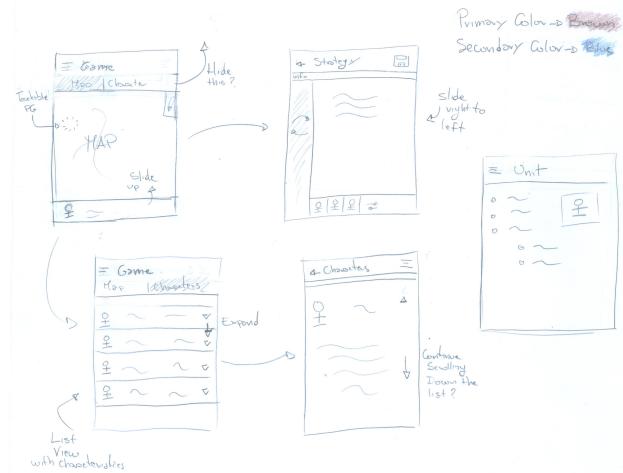


Fig. 3: Primo mock-up dell'applicazione.

Questi schizzi rappresentano una fase molto importante dello sviluppo in quanto mettono le basi per poter iniziare l'implementazione vera e propria del primo prototipo realmente funzionante.

3. Seconda fase: Applicazione

Dopo aver ricavato i requisiti dalla fase di Needfinding e fatto dei primi disegni del software, il lavoro si é concentrato sullo sviluppo vero e proprio di un prototipo: un'applicazione perfettamente funzionante che implementasse tutte le funzionalitá che permettessero di soddisfare gli obiettivi ricavati dai passi precedenti.

3.1. Tecnologie utilizzate

L'intenzione era quella di fare un'app nativa che implementasse al meglio i concetti del material design, per questo è stato scelto lo sviluppo in Java con Android Studio [9] come ambiente di sviluppo. L'obiettivo principale era caricare un salvataggio di gioco e interpretare le informazioni contenute in esso: infatti la maggior parte dei file di Wesnoth segue un linguaggio di mark-up ben specifico chiamato WML, sviluppato interamente per il gioco (non starò a dilungarmi sulla descrizione dei vari tag). Purtroppo i parser presenti nei file sono scritti in C++ o in python, quindi impossibili da implementare dentro un'applicazione Java senza usare JNI, che però avrebbe reso lo sviluppo troppo impegnativo per un semplice prototipo. Per questo è stato utilizzato un parser in python offline per estrapolare le informazioni del salvataggio e metterle dentro un database SQLite, facilmente importabile in Android: in figura 4 è mostrato un breve modello su come è organizzato il database creato. Inoltre, sono stati presi gli asset dal contenuto del gioco e importati nell'applicazione.

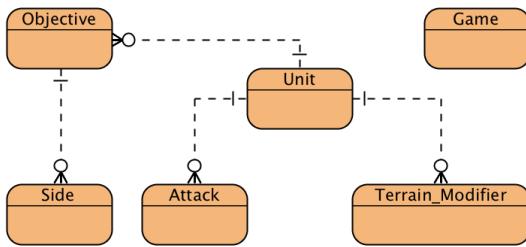


Fig. 4: Modello relazionale del database.

Stesso discorso è stato fatto per il motore di gioco: sarebbe stato inutile perdere tempo per sviluppare una versione java del motore per un semplice prototipo, per questo le mappe relative ai salvataggi sono semplicemente degli screenshot di gioco.

3.2. UML e Model View Controller

In figura 5 è mostrato lo schema UML delle classi create per l'applicazione; è possibile suddividere il grafico in 3 parti:

- Model: il modello rappresenta il livello logico, e le classi che lo definiscono sono quelle che rappresentano i dati più importanti che l'applicazione deve interpretare e modificare. In questo caso si tratta: della classe Character, che definisce tutti gli attributi dei personaggi; della classe Attack, che definisce gli attributi di un attacco: ogni personaggio avrà quindi un array di 1 o più attacchi; la classe General, che racchiude tutte

le informazioni dello scenario di gioco; la classe Notes (in questo caso implementata come una classe statica Singleton di modo che tutte le altre del modello possono accedervi) che racchiude tutte le note eventualmente lasciate dagli utenti; le classi DataBaseAdapter e DataBaseHelper che gestiscono semplicemente il database SQL visto precedentemente. Nelle varie classi sono definiti metodi che modificano le variabili contenute al loro interno;

- View: le view sono rappresentate dai file XML ed eventualmente da qualche custom widget creato per l'occasione; questi file rappresentano la parte grafica dell'applicazione - ciò che vede l'utente - e qui vengono definiti gli elementi interagibili. Il come i dati del modello si devono adattare alle view e il come le view rispondono all'utente sono gestiti dai controller;
- Controller: generalmente le classi controller sono quelle che adattano i dati del modello alle view e gestiscono gli eventi; nel nostro caso questi elementi svolgono la maggior parte del lavoro. Android prevede la definizione di classi dette Activity che rappresentano il main thread di una particolare schermata: è in questa classe che quindi si inizializzano tutti i dati e tutti gli eventuali adapter. Nel nostro caso, la MainActivity (schermata principale) definisce 2 "sub-activity" che in ambito Android vengono chiamate Fragment; questi elementi definiscono le 2 pagine riguardanti la mappa e la lista dei personaggi. Ogni fragment usa degli Adapter per adattare i dati salvati nel modello e inserirli nelle view; inoltre definiscono i metodi di callback degli eventi, creando dei listener che rispondono alle interazioni degli utenti eventualmente modificando le view o direttamente le variabili delle classi modello. Sono inoltre definite altre activity (CharacterActivity e LoginActivity) che si comportano esattamente come la MainActivity, ma hanno un'importanza minore all'interno del sistema;

Il paradigma Model View Controller permette di isolare i componenti chiave della GUI, rendendoli altamente riusabili. Attraverso questa organizzazione si fa sì che il modello non abbia nessun riferimento alle view e viceversa, ma la comunicazione tra le due entità viene gestita interamente dai controller; in questo modo si permette eventualmente di modificare le view o il modello indipendentemente l'una dall'altra, aumentando il grado di riusabilità. Le activity svolgono in un certo senso il ruolo di Controller di Controller, mentre il design pattern Adapter è molto utilizzato in ambito Android in quanto sono molto comuni le liste di elementi o in generale dei dati che devono essere adattati a delle custom view.

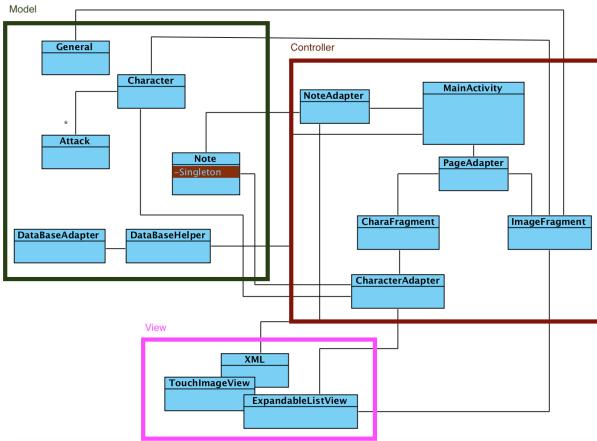
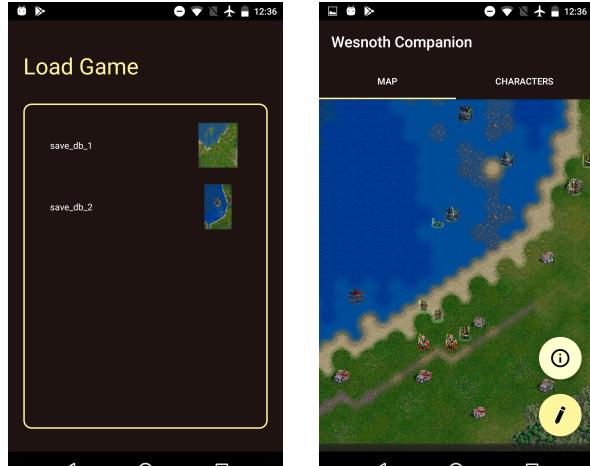


Fig. 5: Diagramma UML dell'applicazione.

3.3. Interfaccia e funzionalità

La prima schermata che si presenta all'utente è la schermata di Login (figura 6a), in cui è possibile selezionare 2 salvataggi fissi della prima campagna elaborati offline come detto precedentemente. L'utente può quindi selezionare uno dei due salvataggi ed entrare nella MainActivity: la prima cosa che viene mostrata è la pagina riguardante la mappa del salvataggio selezionato (figura 6b), con in alto un Tab Selector per poter passare alla pagina della lista dei personaggi (figura 6d); l'uso di questi elementi è un concetto tipico del Material Design. Rimanendo nella schermata della mappa, sono presenti 2 Floating Button - anch'essi tipici del Material: quello più in alto svolge un'azione contestuale alla schermata della mappa, facendo apparire un Bottom Sheet con le informazioni dello scenario; il secondo invece permette di accedere al Drawer che appare da destra a sinistra in cui l'utente può appuntarsi delle note. Quest'ultimo elemento è un elemento persistente anche nella pagina dei personaggi, mentre il primo button appare solamente nella vista della mappa (figura 6c). La mappa è completamente navigabile e zoomabile attraverso le gesture tipiche, mentre i personaggi su di essa sono tutti selezionabili: quando viene fatto un tap su uno di essi si aprirà un Bottom Sheet con il riassunto delle caratteristiche dell'unità, con un area di inserimento testo con la quale l'utente può scrivere delle note direttamente sul personaggio: queste andranno ad aggiornare il Drawer delle note generali, con il colore relativo all'esercito di appartenenza (figure 7a e 7b). Selezionando invece un personaggio dalla schermata relativa, si espanderà nella lista una sezione in cui è possibile vedere le sue caratteristiche (figura 7c) e la solita area di input per le note, più un tasto con il quale si può andare a vedere il personaggio nella mappa. Nella parte bassa di questa sezione si trova un link che indirizza alla pagina descrittiva dell'unità



(a)

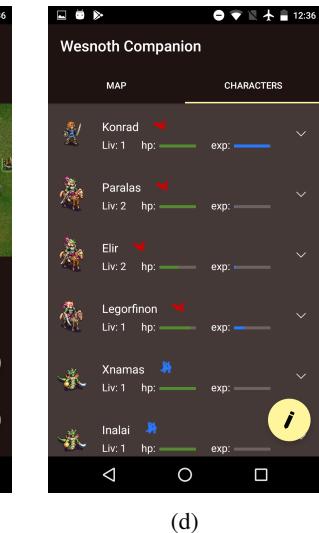
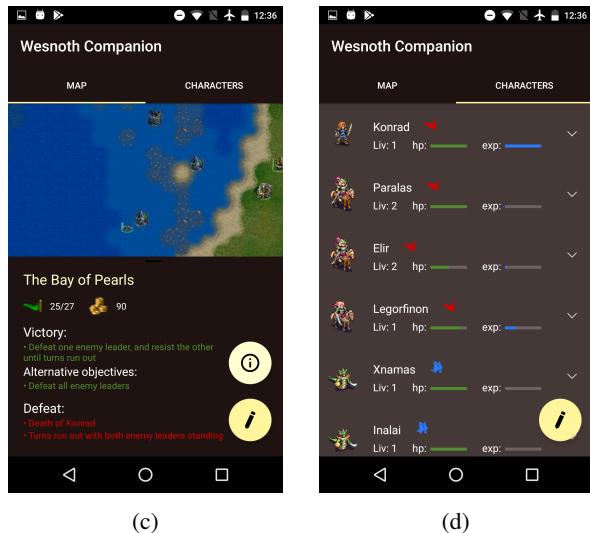


Fig. 6: Vari screenshot dell'applicazione: (a) activity di login, (b) schermata della mappa, (c) informazioni relative allo scenario, (d) lista dei personaggi.

- la CharacterActivity (figura 7d): qui l'utente può visionare tutte le informazioni dettagliate della classe e può navigare in quelle successive se e solo se sono già state scritte. La lista comprende tutti i personaggi di tutti gli eserciti, sia nemici che alleati, e l'appartenenza a un relativo side è definita dal colore posto vicino al nome. È possibile eliminare le note direttamente dalla scheda del personaggio, oppure nel drawer relativo: ogni modifica fatta verrà automaticamente notificata a tutti gli elementi del modello per poter aggiornare tutte le view di conseguenza (tutto questo tramite i controller). Grande importanza viene riposta nelle animazioni e nella reattività dei comandi, come vuole il Material Design: ogni animazione è fatta in modo di essere breve e con l'obiettivo di attirare l'attenzione dell'utente

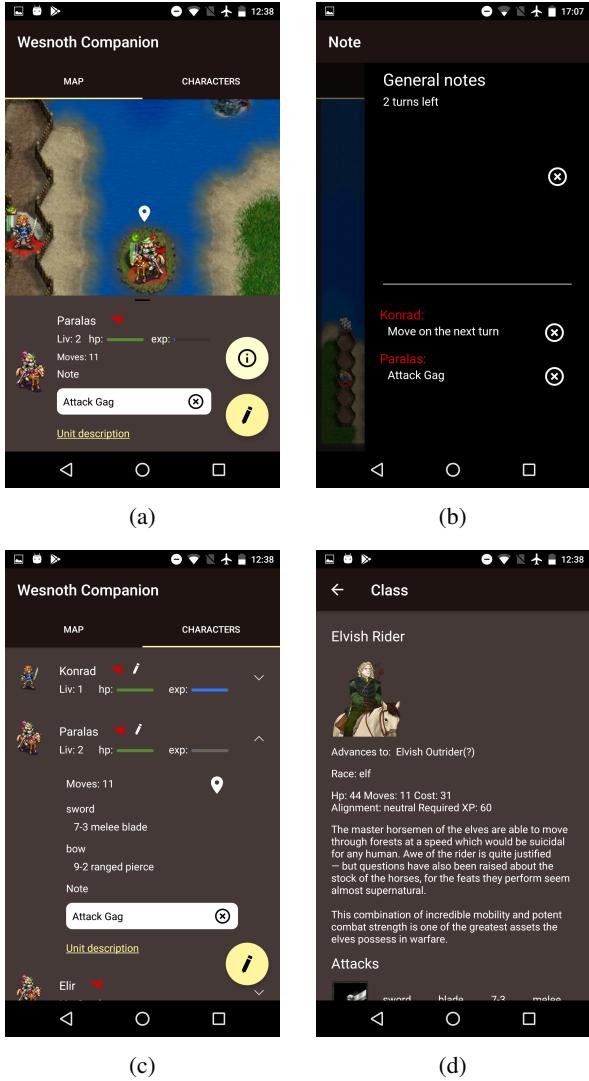


Fig. 7: Altri screenshot dell'applicazione: (a) sheet con le informazioni dei personaggi, (b) drawer con le note (c), lista dei personaggi espansa, (d) descrizione dettagliata dell'unità.

su un determinato punto dello schermo; infatti l'occhio umano è molto sensibile ai movimenti provenienti dalla parte più esterna dell'occhio. Per migliorare questo effetto, il Material prevede di definire delle animazioni che siano più naturali possibili, descrivendo una curva di accelerazione/decelerazione piuttosto che avere un'interpolazione lineare dei punti in cui si muove l'elemento.

4. Terza fase: Test di Usabilità

L'ultima fase dello sviluppo si concentrava su una valutazione del prototipo tramite un'analisi dell'usabilità. I test sono stati effettuati scegliendo gli stessi intervistati che ave-

vano provato Wesnoth nella fase di Needfinding, persone che avessero una conoscenza del gioco necessaria per avere un'idea dell'applicazione: i candidati erano quindi persone tra 18 e 26 anni di ogni genere. Ogni test è stato svolto in ambienti familiari come bar e prevedeva una prima fase di prova dell'applicazione: ai candidati è stato consegnato un dispositivo Android sul quale era già pre-installata la companion app e si è lasciato libero l'utente di provare tutti gli elementi per un periodo di ambientamento. Dopo questa prima fase, è stato effettuato un test scriptato dove sono stati dati dei compiti da svolgere:

- Goal 1: caricare il salvataggio 2 e navigare la mappa;
- Goal 2: selezionare sulla mappa il personaggio Konrad e scrivere una nota sulla sua scheda;
- Goal 3: selezionare nella lista dei personaggi il nemico Gag e scrivere una nota sulla sua scheda;
- Goal 4: selezionare l'alleato Legorfinon, localizzarlo nella mappa e scrivere una nota sulla sua scheda;
- Goal 5: scrivere delle note generali nel drawer;
- Goal 6: eliminare tutte le note;

Sia durante la prima fase sia durante la risoluzione degli obiettivi è stato chiesto all'utente di parlare ad alta voce per raccogliere le sue impressioni. Alla fine della seconda fase è stato chiesto di rispondere ad un questionario: questo era composto da 18 domande di tipo SEQ con una scala che va da 1 a 7, dove 1 indica fortemente disaccordo e 7 fortemente d'accordo. Le domande potevano essere di ordine generale o più tecnico e nella tabella 1 sono riassunti i risultati dei test. Per un'analisi quantitativa dell'app è stato misurato sia il tempo medio del primo periodo di ambientamento sia il tempo medio di completamento degli obiettivi, con risultati più che soddisfacenti visibili nella tabella 2.

	Prima fase	Seconda fase
Tempo medio	3 minuti	2 minuti

Tab. 2: Test quantitativo dell'applicazione.

I commenti più comuni e interessanti - riscontrabili anche nelle risposte alle domande - sono stati:

- L'applicazione è visivamente piacevole e rispecchia i colori e il feeling del gioco originale. Inoltre le animazioni sono risultate gradevoli e aiutavano a mantenere il flow logico dell'applicazione e a raggiungere gli obiettivi;
- Dopo la prima fase di ambientamento, in cui potevano esserci delle piccole difficoltà, l'applicazione risulta molto intuitiva e logica e "non c'è differenza tra chi la usa da un giorno o da molti anni";

N	Domande	Media	σ
1	L'applicazione è visivamente gradevole e rispecchia il "look and feel" del gioco	6.37	0.48
2	Il significato delle icone e dei pulsanti è chiaro	6.5	0.50
3	Le animazioni sono fluide e piacevoli	6.62	0.48
4	Il posizionamento dei pulsanti è sbagliato e illogico	1.25	0.43
5	Le etichette sono chiare e riflettono quello che fanno	6.25	0.66
6	L'app presenta tanti errori	1.75	0.66
7	Le gesture sono intuitive e naturali	6.5	0.70
8	È difficile navigare la mappa	2	1.22
9	La risposta quando un personaggio è selezionato sulla mappa è rapida	6.75	0.43
10	La lista dei personaggi è mostrata correttamente e l'interazione con essa è piacevole	6.37	0.69
11	È difficile trovare un personaggio sulla mappa	1.25	0.66
12	È facile scrivere delle note (sia generali o direttamente sulla scheda del personaggio)	6.87	0.33
13	Il modo in cui vengono mostrate nella mappa le informazioni dei personaggi è soddisfacente	6.37	0.69
14	Il modo in cui vengono mostrate nella mappa le informazioni dello scenario è soddisfacente	6	0.70
15	È difficile cancellare una nota	1.00	0.00
16	Le descrizioni delle unità sono difficili da trovare o da leggere	1.87	1.26
18	In generale il sistema è gradevole	6.5	0.5

Tab. 1: Tabella dei risultati del test di usabilità dove σ è la deviazione standard. Come è possibile vedere, gli utenti sono rimasti abbastanza soddisfatti della companion.

- Ci sono comunque dei piccoli problemi che possono essere risolti: ad esempio è stata trovata un po' di difficoltà nel chiudere la tastiera; la navigazione della mappa non è risultata ottimale a tutti i tester per via delle gesture per cambiare pagina (domanda 8); è stata trovata difficoltà nel tornare dalla lista dei personaggi alla mappa, in quanto alcuni tester premevano il tasto "back" che fa uscire dall'applicazione, mentre è atteso che cambino pagina dal Tab Selector posto in alto; le descrizioni delle unità non sono risultate sufficienti a tutti i tester (domanda 16);
- I candidati hanno confermato l'utilità di una companion app di questo genere, soprattutto per un gioco altamente strategico e potenzialmente molto complicato come Wesnoth: le partite potrebbero durare molto tempo, e l'idea di avere un modo per prendere delle note strategiche al di fuori del contesto di gioco è stata trovata molto interessante.

5. Conclusioni e miglioramenti

In questo progetto è stata quindi sviluppata e testata una companion app per il gioco strategico The Battle for Wesnoth, seguendo tutte le fasi della Human Computer Interaction: needfinding, implementazione e test di usabilità. È stata quindi creata un'app nativa Android che seguisse i principi del Material Design. Dalla fase di testing l'applicazione è risultata piacevole da vedere e da utilizzare, intuitiva e logica, che potrebbe rendere l'esperienza di gioco più completa; non sono inoltre stati evidenziati prob-

lemi molto evidenti se non qualche piccolo bug facilmente risolvibile. I miglioramenti suggeriti sono stati:

- L'aggiunta di più informazioni all'interno delle schede del personaggio o direttamente nella mappa, che però potrebbe portare ad una visione non ottimale di essa;
- Un modo alternativo di scrivere le note, ad esempio la possibilità di disegnare sulla mappa, che però potrebbe andare contro alle gesture per la navigazione;
- Un metodo per visualizzare meglio il movimento possibile dei personaggi.

Come futuri sviluppi sarebbe interessante approfondire e studiare i suggerimenti dei tester, con l'aggiunta di una conversione del parser da python a java e l'importazione del motore di gioco per generare la mappa originale del salvataggio. In conclusione comunque i candidati sono rimasti molto soddisfatti dal sistema.

Riferimenti

- [1] The Battle for Wesnoth;
<https://www.wesnoth.org>
- [2] The Battle for Wesnoth per Android;
<https://play.google.com/store/apps/details?id=it.alessandropira.wesnoth112&hl=it>
- [3] The Battle for Wesnoth per iOS;
<https://itunes.apple.com/us/app/battle-for-wesnoth/id575239775?mt=8>

- [4] X-COM, il gioco;
<https://xcom.com>
- [5] Heroes of Might and Magic VII;
<https://www.ubisoft.com/it-it/game/might-and-magic-heroes-7/>
- [6] Civilization VI;
<https://www.civilization.com>
- [7] Sistema operativo Android;
<https://www.android.com>
- [8] Material Design by Google;
<https://material.io>
- [9] Android Studio;
<https://developer.android.com/studio/index.html>