Federico Barbero 20024119
Simone Gattini 20025432

Dipartimento di Scienze ed Innovazione Tecnologica

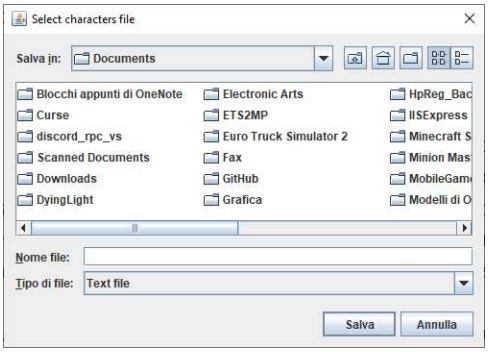
Ingegneria del Software

1.1 Requisiti non funzionali

RNF01: l'interfaccia utente sarà grafica, semplice e intuitiva.
RNF02: il cliente interagirà solo l'interfaccia utente.
RNF03: l'applicazione sarà veloce nel rispondere alle richieste dell'utilizzatore.
RNF04: implementazioni successive per future aggiunte dovranno poter essere aggiunte rapidamente senza revisioni complete.
RNF05: i file sono salvati in maniera ordinata, nella posizione scelta dall'utente.
RNF06: l'applicazione dovrà essere leggera e facilmente gestibile dal sistema.
RNF07: il sistema sarà implementato con in linguaggio Java.

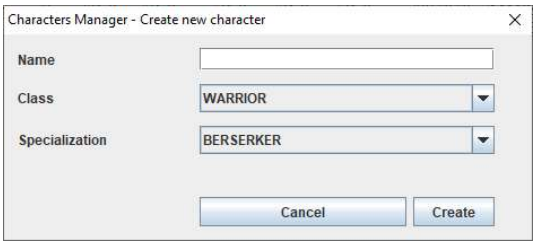
1.2 Requisiti Funzionali

1.2.1 **RF01**: **salvataggio dati**



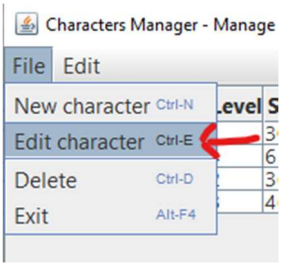
Il servizio di salvataggio dati è il primo che viene utilizzato dall'utente, senza questo passaggio fondamentale l'applicazione non potrà formalmente gestire i dati. Verrà controllato se l'utente sceglie correttamente la posizione di destinazione dei file. Se l'utente non sceglie la posizione dei file, gli verrà richiesto successivamente, anche in caso di inserimento ripetuto. Nel caso in cui l'utente salti tutti questi passaggi può risolvere andando nelle impostazioni impostando i file sempre scegliendone il percorso.

1.2.2 **RF02**: **creazione del personaggio**



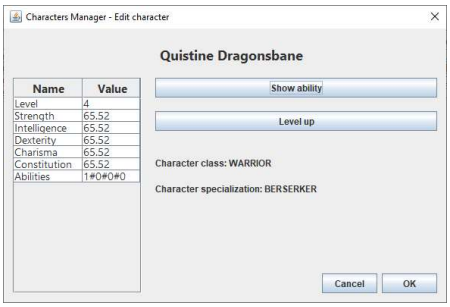
Il servizio di creazione del personaggio gestisce le informazioni iniziali del personaggio, ne imposta i valori iniziali predefiniti ed integra nella tabella il nuovo personaggio appena creato. Questa interfaccia richiede all'utente di scegliere il nome del personaggio, la classe e la specializzazione del personaggio stesso. La sottoclasse è validata automaticamente in base alla scelta della classe del personaggio. Una volta conclusa l'operazione, l'utente potrà selezionare e modificare il personaggio.

1.2.2 **RF02**: **modifica del personaggio**



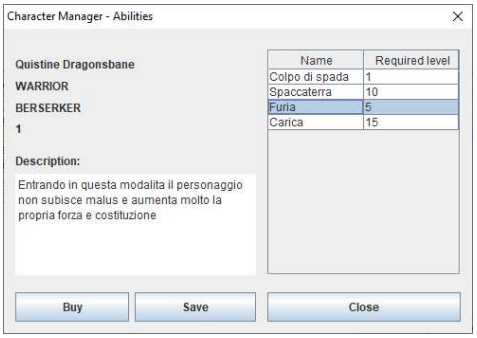
Dopo aver selezionato il personaggio nella tabella principale, l'utente utilizzando l'apposito pulsante al percorso *File >> Edit Character* (disponibile anche con la scorciatoia da tastiera *CTRL + E*) è possibile modificare dettagli e specifiche del singolo personaggio.
L'azione eseguita apre così una nuova finestra per la gestione del personaggio.
È possibile inoltre cancellare il personaggio selezionato con l'apposito pulsante *File >> Delete* o con la scorciatoia *CTRL + D*)

Federico Barbero 20024119
Simone Gattini 20025432



1.2.4 **RF02**: **modifica del personaggio e livellamento**
L'utente ha a disposizione un ulteriore interfaccia per visualizzare i dettagli del personaggio scelto. Questi
dati sono contenuti in un file, che conterrà le specifiche del personaggio, le abilità acquisite ed il nome del personaggio.
Attraverso il pulsante Level up si aumentano le statistiche del personaggio stesso, facendo in modo che tutte le statistiche mutino sulla base di un calcolo automatico implementato nel codice. Il calcolo fa uso di un moltiplicatore non dichiarato che gestisce in autonomia le specifiche.
Attraverso il pulsante *Show ability* invece si accede ad un'altra interfaccia per modificare le abilità.
La finestra fa distinzione sui tasti di chiusura: il pulsante OK conferma le modifiche, gli altri tasti no.

1.2.5 **RF05**: **modifica delle abilità**



Dopo aver selezionato *Show ability*, l'utente potrà scegliere tramite la finestra *Abilities* quali abilità inserire sul personaggio scelto. Per acquisire una nuova abilità il personaggio andrà livellato e riceverà punti abilità da spendere.
Le abilità vanno soddisfatte con il livello richiesto del personaggio, se il personaggio non possiede il livello richiesto o non possiede punti abilità sufficienti, l'abilità non può essere acquistata.
Anche in questa finestra i pulsanti fanno la differenza: il pulsante Save comunica le modifiche effettuate, gli altri tasti no.

1.3 Studio di fattibilità

Dopo una prima fase di studio, si è arrivati alla conclusione che per realizzare il software in questione le risorse a nostra disposizione sono più che sufficienti. Non è previsto l'acquisto di ulteriori componenti tecnologici. I tempi e i costi di richiesta sono ritenuti accettabili, lo sviluppo è previsto con un mese di lavoro circa. Queste informazioni sono basate anche sulla documentazione di altri progetti simili.
Per il buon funzionamento del software è vivamente sconsigliata la modifica manuale dei file forniti, si fornisce comunque, a pagamento, assistenza tecnica non manutentiva per eventuali errori causati dagli utenti.

1.4 Collaudo

Il software è stato collaudato sulla base della TDD o Test-Driven-Development. È stato inoltre collaudato manualmente con un programma di beta-testing gestito. Il programma di beta-testing è stato gestito attraverso una community di utenti esperti, programmatori e manager aziendali, così da assicurarne il corretto funzionamento durante l'esecuzione utente.

1.5 Manutenzione

È previsto un vasto supporto all'applicazione, minimo 5 anni, con bug fixes, nuove features implementate su richiesta e release di correzione grafica con ottimizzazione.

Federico Barbero 20024119
Simone Gattini 20025432

1.6 Form Template

a) Validità:

Applicando la strategia **white box** abbiamo cooperato con il committente del software. Sfruttando questa strategia lo sviluppo dell'applicazione è stato più agile e mirato alle richieste effettive del cliente, correggendo e migliorando, eventualmente, determinate features non gradite dal committente stesso.

La necessità di un interfacciamento semplice ed intuitivo per l'utente finale ha richiesto molto lavoro, soprattutto per alleggerire il carico di lavoro all'utente, rendendo più compatta e semplice la gestione dei dati. L'usabilità dell'applicazione viene rispettata proprio dalla GUI (o Graphic User Interface) che la rende di facile interpretazione e semplice da utilizzare.

I personaggi sono elencati in modo da essere riconosciuti facilmente ed ordinati in modo alfabetico, la tabella principale è di facile lettura, anche grazie ai vari titoli delle colonne che ottimizzano lettura e valori.

b) Svolgimenti:

- Svolgimenti successivi: è possibile inserire, tramite un database, un sito web con autenticazione personale che gestisca i dati che implementi le stesse funzionalità dell'applicazione desktop, senza sostituirla con la possibilità di aggiornare le informazioni dell'utente e dei personaggi da qualsiasi postazione tramite l'account personale.

- Svolgimento attuale: all'avvio, l'applicazione controlla che i file di lavoro siano correttamente posizionati, se il risultato da esito positivo vengono aperti e letti automaticamente, altrimenti vengono segnalati malfunzionamenti all'utente che può scegliere la nuova/vecchia posizione del file.

c) Metodo:

Il progetto è stato implementato sfruttando la metodologia "Top-Down". Partendo dal modulo più complesso, ossia il personaggio, siamo scesi sino a trasformarlo in una lista di stringhe, che ne permette la facilità di modifica, implementazione, gestione e salvataggio. Il modulo base su cui il software lavora praticamente sempre è il gestore dei file, che tramite i vari gestori trasforma le stringhe in tutti i vari attributi del personaggio.

Il progetto sfrutta un paio di pattern che lo rendono molto efficace e semplice da usare e mantenere:

- Strategy: poiché il gestionale è stato fatto per seguire una sequenza precisa di azioni da fare, il programmatore induce l'utente a seguire un pattern di azioni predefinito, molto simile al pattern "Command".
- State: utilizzato per definire i vari stati di modifica in cui il personaggio si trova, per esempio durante la modifica delle abilità associate o mentre viene fatto salire di livello. Per ogni vista viene associato un tipo di controllo specifico che ne definisce correttamente il comportamento da seguire durante la modifica.