Documentazione per progetto di ingegneria del software

Tomasoni Francesco (1080980) – Maccari Luca (1081951)

Anno accademico 2023-2024

1. **Project plan**
   1. Introduzione

Il progetto prevede la realizzazione in Java di una versione “a modo mio” del noto gioco Space Invader. L'obiettivo principale è dimostrare di saper applicare i concetti, le metodologie e tecniche appreso durante il corso.

Il giocatore ha lo scopo di distruggere le navicelle aliene che lentamente si avvicinano alla sua navicella, se il giocatore distruggerà tutte le navicelle aliene passerà al livello successivo dove aumenterà la difficoltà.

Il punteggio finale è dato dal numero di livelli superati.

Se la navicella del giocatore viene distrutta, il gioco finisce.

Questa versione di gioco oltre ad avere tutte le funzionalità del gioco originale dovrà implementarne ulteriori, tra cui:

- possibilità dell’utente di cambiare tipo di arma

- Il gioco dovrà supportare più lingue

- Il gioco dovrà essere disponibile per Windows, Mac e Linux

//TODO

Il team è composto da: Tomasoni Francesco e Maccari Luca.

* 1. **Modello di processo**

Il modello da noi utilizzato è l’Extreme Programming, ovvero un modello di tipo agile.  
Per sviluppare al meglio il nostro progetto abbiamo deciso di utilizzare questo modello in quanto rispettava questi requisiti:

|  |  |
| --- | --- |
| XP Practice | Descrizione |
| Pair programming | Tutto il codice viene scritto da due programmatori sulla stessa “macchina”. |
| Collective ownership | Ognuno può modificare il codice quando vuole. |
| Small release | Il codice deve essere funzionante a fine di ogni giornata, attraverso piccole modifiche. |
| Testing | Ogni modifica al codice viene testata all’istante. |

**1.3 Organizzazione del progetto**

Le persone coinvolte nella progettazione del gioco inizialmente sono quelle che compongono il team.

In seguito alla pubblicazione, i giocatori potranno prendere parte allo sviluppo suggerendo nuove funzionalità o segnalando bug, o, se lo desiderano, potranno contribuire direttamente allo sviluppo facendo un *fork* del repository, implementando le loro modifiche e aprendo una *pull* *request*. Affinché le modifiche entrino a far parte del gioco dovranno essere revisionate e approvate dal team di sviluppo.

**1.4 Standards, linee guida e procedure**

Verrà utilizzato java come linguaggio di programmazione

Gli IDE utilizzati sono VSC e Eclipse.

Per gestire le riunioni del team e assegnazione di compiti viene utilizzato Discord.

**1.5 Attività di gestione**

Si è deciso di organizzare lo sviluppo seguendo lo schema dell’*Extreme programming* in quanto permette di migliorare la qualità del codice e approcciarsi al cambiamento dei requisiti in modo più responsivo.

Si è deciso di applicare l'approccio del *pair programming* per la scrittura del codice, con il quale si hanno coppie di sviluppatori che lavorano insieme su un singolo frammento di codice e si dividono il lavoro di scrittura e revisione in tempo reale.

Si è deciso di accordare incontri settimanali in base al tempo a disposizione del team sono di breve durata, all'incirca 30 minuti, durante i quali possono essere effettuate delle sessioni di brain storming per trovare nuove idee o cercare soluzioni a problemi riscontrati.

Al termine del meeting, ci si accorda sui compiti da portare avanti durante le settimane successive.

**1.6 Rischi**

Il rischio principale è di non consegnare entro la deadline il progetto oppure di consegnare qualcosa di non perfettamente funzionante.

**1.7 Membri**

La modalità di lavoro \*agile\* fa si che non ci sia una netta distinzione tra i ruoli nel gruppo di lavoro.

Viste le regole date dall’XP abbiamo evitato la suddivisione in ruoli del team (manager…), in modo da avere la stessa importanza.

Grazie ai differenti skill personali degli sviluppatori coinvolti entrambi i membri del team hanno dato un contributo attivo al progetto sfruttando al meglio le proprie competenze.

1.8 Metodi e tecniche

//TODO

/\*

* **StarUML** per quanto riguarda il design, nello specifico: Use Case Diagram, Class Diagram, Sequence Diagram e Activity Diagram e State Machine Diagram.
* Per la coordinazione del lavoro e dei file abbiamo utilizzato **GitHub**
* **Eclipse IDE** per la realizzazione e sviluppo del codice Java.

\*/

//loro hanno messo diagramma use case

**1.9 Garanzie di qualità**

Si punta a sviluppare un software che rispetti i parametri di qualità indicati dal modello di McCall:

* Correttezza
* Affidabilità
* Robustezza
* Integrità
* Usabilità

Per garantire la sicurezza degli utenti sono state usate librerie moderne e molto diffuse, aggiornate frequentemente e poco inclini ad avere vulnerabilità.

Per verificare la corretta funzione del programma, nella fase di testing abbiamo testato ogni possibile scenario di gioco immaginabile. Inoltre, per controllare la qualità abbiamo condiviso il gioco a terzi per verificare la giocabilità e i possibili errori (bug) del software.

**1.10 Package di lavoro**

Alcuni dei sotto progetti (work package) che sono stati definiti a priori.  
In viste della natura agile del progetto, questi work-package saranno estesi e modificati o evoluti nelle diverse iterazioni del life-cycle:

• Fase di design di schemi UML  
• Implementare l’applicazione con un’interfaccia grafica (front-end)   
• Uso di un database  
• Fare il testing sul prodotto

**1.11 Risorse**

Le risorse sono state le nostre conoscenze pregresse e acquisite durante il corso.

A livello hardware e software sono stati utilizzati dei PC con caratteristiche standard per l’utilizzo di Eclipse e di cloud-server.

**1.12 Budget e pianificazione**

Immagine che contiene diagramma, Piano, linea, testo

Descrizione generata automaticamenteIl budget finanziario è stato di zero euro. Possiamo considerare come budget il tempo speso, utilizzando il **PERT** (Programming Evolution and Review Techniques).  
Questa tecnica rappresenta le attività in una rete di nodi, nota come “rete di attività su nodo”. Ad ogni nodo viene assegnato un peso il quale indica la durata dell’attività corrispondente, calcolato usando la formula: (tempo ottimistico + (4 volte più probabile) + tempo pessimista) / 6  
I collegamenti con le frecce indicano che un’attività non può iniziare prima del completamente dell’attività precedente.

1.13 Cambiamenti

Oltre alle features base incluse nella versione alpha, il team ha fissato come obiettivo l'aggiunta di funzionalità e nuove modalità di gioco per fornire un gioco più completo e variegato.

//TODO

- Traduzione del software in altre lingue oltre all'inglese e all'italiano

* 1. **Consegna**

Il project plan verrà consegnato entro il 19 Dicembre.

IL progetto entro la data del primo appello di esame (19 Gennaio)