

# Tank Game

Davide Michelotti, Marco Ghirini, Riccardo Strina

con l’aiuto del Prof. Paolo Ollari

Sommario

[Tank Game 1](#_Toc513403179)

[Sommario 2](#_Toc513403180)

[1 INTRODUZIONE 3](#_Toc513403181)

[2 PRESENTAZIONE DELL’ARGOMENTO 3](#_Toc513403182)

[2.1 - giustificazione della scelta dell’argomento 3](#_Toc513403183)

[2.2 - problemi incontrati nella realizzazione del lavoro 3](#_Toc513403184)

[2.3 - metodologia seguita 3](#_Toc513403185)

[3 SVILUPPO 4](#_Toc513403186)

[3.1 C# 4](#_Toc513403187)

[3.2 UDP 4](#_Toc513403188)

[3.3 GitHub 6](#_Toc513403189)

[4 STRUTTURA 7](#_Toc513403190)

[4.1 schema concettuale 7](#_Toc513403191)

[4.2 schema videate 8](#_Toc513403192)

[5 CONCLUSIONE 8](#_Toc513403193)

[6 APPENDICI 8](#_Toc513403194)

[7 BIBLIOGRAFIA 8](#_Toc513403195)

# 1 INTRODUZIONE

Il progetto consiste in un’applicazione desktop, più precisamente un gioco multiplayer, in cui da 2 a 20 giocatori si sfideranno con dei carrarmati 2D, sparandosi all’interno di una mappa.

Il progetto è formato da un solo applicativo, non vi è la presenza di un server vero e proprio, in quanto un client designato svolgerà la funzione di host, comportandosi sia da client che da server. D’ora in poi quindi, quando useremo il termine “server”, faremo riferimento ad un normale PC, che svolge solo la funzione di server, non è infatti, una macchina server. Vi è quindi una prima fase di negoziazione della connessione, in cui, tramite un algoritmo, un client si elegge server, se non trova un server già attivo

Il linguaggio utilizzato per sviluppare l’applicazione è il C#, con l’ambiente di sviluppo Visual Studio; per quanto riguarda il protocollo di comunicazione si è scelto di utilizzare UDP, approfondito nella materia “Sistemi e Reti” durante l’anno, invece di TCP, che risultava più complesso.

# 2 PRESENTAZIONE DELL’ARGOMENTO

### 2.1 - giustificazione della scelta dell’argomento

Il progetto nasce dall’esigenza del prof. Paolo Ollari di avere un programma da utilizzare durante gli OpenDay dell’ITIS, col fine di intrattenere i ragazzi delle medie e di mostrare loro uno dei tanti argomenti che si affrontano durante il triennio, nell’indirizzo informatico.

Il miglior programma per impressionare tale target di studenti è un videogioco, con elementi grafici e possibilità di giocare contro altri giocatori in tempo reale.

È da questo che nasce il progetto “Tank Game”.

### 2.2 - problemi incontrati nella realizzazione del lavoro

Il problema principale che è stato riscontrato riguardi i ritardi e i rallentamenti della grafica o la difficoltà di gestione dei vari elementi di gioco (carrarmati e proiettili). Inizialmente si era pensato di usare solo ciò che metteva a disposizione Visual Studio, infine si è deciso di utilizzare un framework per la gestione grafica, in quanto forniva più funzionalità e rendeva il tutto più fluido.

### 2.3 - metodologia seguita

Durante lo sviluppo del software si è scelto di utilizzare la metodologia di Ingegneria del Software chiamata “Incremental by Test”. Questa tecnica consiste nel creare un progetto per ogni “problema”, cioè per ogni parte, in modo da dividere tra i membri del gruppo il lavoro e concentrarsi su piccoli problemi. Una volta risolti i problemi più elementari si può procedere con il merge di tutti i progetti e giungere a quello finale.

Questa metodologia si abbina alla scelta di utilizzare GitHub come servizio di hosting del progetto.

In particolare il lavoro è stato suddiviso nel seguente modo:

* Davide Michelotti: grafica
* Marco Ghirini: documentazione, GitHub, grafica
* Riccardo Strina: protocollo UDP

# 3 SVILUPPO

Di seguito sono elencati e descritti il linguaggio di programmazione utilizzato, il protocollo di trasporto dati e il servizio di hosting del progetto.

## 3.1 C#

Il C# è un linguaggio di programmazione orientato agli oggetti sviluppato da Microsoft all'interno dell'iniziativa .NET. La sintassi e struttura del C# prendono spunto da vari linguaggi nati precedentemente, in particolare Java.

1 Logo ambiente di sviluppo

Questo linguaggio è l’unico utilizzato durante lo sviluppo, in quanto approfondito durante l’anno scolastico e ritenuto il più idoneo.

Come ambiente di sviluppo è stato utilizzato l’IDE Visual Studio(Versioni 2015 e 2017).

E’ stato, inoltre, aggiunto il pacchetto aggiuntivo di GitHub, per la gestione direttamente dall’ambiente di sviluppo del versioning del programma.

## 3.2 UDP

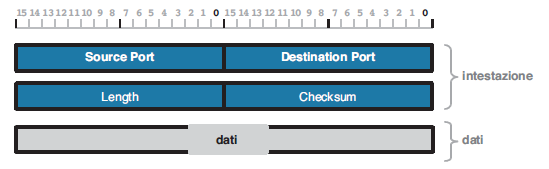
UDP (User Datagram Protocol) è un protocollo di trasporto utilizzato nella rete TCP/IP, di tipo non connesso. Tra le funzioni tipiche di un livello di trasporto, UDP gestisce solo l’indirizzamento, la multiplazione e il controllo dell’errore sul singolo pacchetto, mentre lascia al livello superiore (Applicazione) tutti gli altri compiti. Siccome è un protocollo senza connessione, la sua affidabilità non è garantita, al contrario è molto elevata la velocità.

Questo protocollo consente una comunicazione full-duplex, è inoltre di tipo peer-to-peer (P2P), ovvero ognuno dei due lati della comunicazione ha le medesime capacità.

Il fatto che il programma sia client/server e il protocollo P2P, non è un errore, ma bensì il protocollo è stato implementato e gestito dal gioco in maniera tale che risulti un’architettura client/server.

La scelta di utilizzare UDP, invece di TCP è dovuta dal fatto che è stato studiato approfonditamente durante l’anno. Inoltre UDP si caratterizza per la grande velocità, fattore determinante per la riuscita del gioco discapito di una minor sicurezza che nel nostro caso non era fondamentale. Inoltre essendo pensato per essere utilizzato nella LAN scolastica, il problema viene meno, in quanto le reti LAN sono considerate sicure.

Schema di un pacchetto UDP



Per effettuare la comunicare tra client e server è stato necessario creare un protocollo nostro, che racchiudesse tutte i possibili pacchetti che sarebbe stato necessario analizzare durante lo svolgimento del gioco e la creazione delle partita. Durante lo sviluppo è stato necessario effettuare modifiche, perché ci si è accorti che servivano ulteriori pacchetti, che non erano stati “concepiti” durante una prima analisi. Tutti i pacchetti sotto forma di stringa, e un pacchetto contiene più informazioni, le quali sono divise da un separatore che in questo caso è il ‘;’.

Fattore che abbiamo dovuto tenere in considerazione è che due client non comunicano mai tra di loro, le comunicazioni avvengono sempre tra client e server.

Di seguito ecco i vari pacchetti, suddivisi per categoria:

* Pacchetti di connessione:

**DESTINATARIO <- MITTENTE PACCHETTO**

**BROADCAST <- CLIENT l**

Ricerca di un server, “l” sta per “look”.

**INDIRIZZO <- SERVER id\_Server**

Risposta alla ricerca del server, viene restituito l’id del server.

**INDIRIZZO <- CLIENT r;username**

Ricerca della partita, ‘r’ sta per “request”, mentre ‘username’ è il nome del giocatore.

**INDIRIZZO <- SERVER id;acc/ ref**

Risposta del server al pacchetto precedente, “acc” sta per accettato, “ref” per rifutato.

**INDIRIZZO <- SERVER id;str**

Start della partita.

**INDIRIZZO <- SERVER idClient;pos;x;y;z**

“pos” indica che questo pacchetto contiene le posizioni, ‘x’,’y’,’z’ sono, appunto, le coordinate della posizione iniziale del carro armato.

* Pacchetti gioco:

**INDIRIZZO <- CLIENT for/bck**

“for” sta per avanti, “bck” per indietro, questo pacchetto gestisce l’avanzamento del carro armato.

**INDIRIZZO <- SERVER idClient;pos;x;y;z**

Analogo al pacchetto inviato dal server al client, contente la posizione iniziale del carro armato.

**INDIRIZZO <- CLIENT fire**

Il client comunica che ha sparato.

**SERVER <- INDIRIZZO blt;idBullet,pos;x;y;z**

Il pacchetto contiene le coordinate del proiettile.

**INDIRIZZO <- SERVER blt;idBullet;idClient/0**

In base al fatto che il proiettile abbia colpito o meno un giocatore, viene inviato il suo id, o ‘0’.

**INDIRIZZO <- CLIENT dct**

Il client avvisa il server che si è disconnesso.

**INDIRIZZO <- SERVER idClient;dct**

Il server avvisa gli altri client che è avvenuta una disconnessione.

**INDIRIZZO <- SERVER** idClient;win

Un giocatore ha vinto.

**INDIRIZZO <- SERVER** idClient;USERNAME

Viene richiesto l’username tramite un id.

## 3.3 GitHub

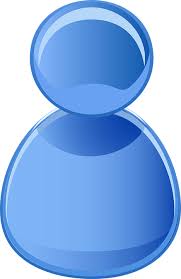
GitHub è un servizio di hosting per progetti software, basato sul software open source Git. Offre piani hosting sia a pagamento, per i progetti software privati, sia gratuiti, per i progetti software open source.

Git offre la possibilità di creare più versioni aggiornate dello stesso progetto, dette “commit”, con l’aggiunta di un commento, che serve a far capire che modifiche sono state fatte rispetto alla versione precedente.

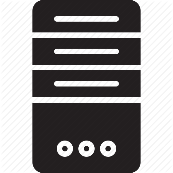
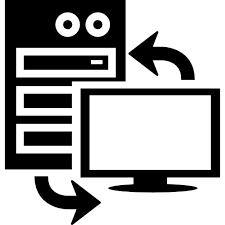
E’ possibile, inoltre, creare più rami di un progetto, ovvero parti distinte di uno stesso progetto o modifiche radicali. Nel nostro caso è stato creato un ramo “grafica”, che contiene tutti i progetti sulla creazione della grafica e della logica di gioco, un ramo “protocollo”, che contiene tutti i progetti sulla comunicazione di rete, infine il ramo di default “master”(ovvero principale), è stato utilizzato per le varie versioni del progetto completo. Una volta completati i due sotto rami (protocollo e grafica) è stato effettuato il merge all’interno del ramo principale, il master.

# 4 STRUTTURA

## 4.1 schema concettuale

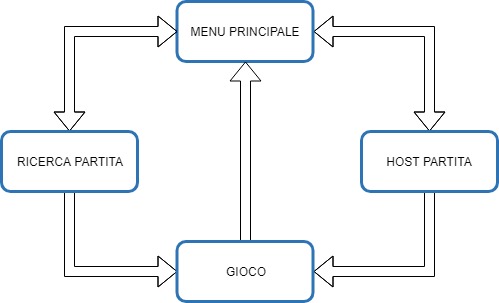






1. Gli utenti avviano l’applicazione desktop “Tank game”, e per prima cosa scelgono il colore del loro carro armato.
2. Una volta fatto ciò, si avvia la ricerca di un server che ospiti una partita.
3. Ci sono due possibilità per iniziare la partita:
4. Un client ha già assunto la funzione di server, quindi ci si collega ad esso;
5. Non ci sono server disponibili, quindi il client decide di assumere lui la funzione di server e ospitare la partita

## 4.2 schema videate



# 5 CONCLUSIONE

FINE PROGETTO

# 6 APPENDICI

FINE PROGETTO

# 7 BIBLIOGRAFIA

Per lo sviluppo sono stati usati i seguenti siti, contenenti guide per lo sviluppo

* [Stack Overflow](https://stackoverflow.com/)
* [CodeProject](https://www.codeproject.com)

A livello consultivo sono stati usati i seguenti siti

• [Wikipedia](https://it.wikipedia.org/wiki/Pagina_principale)

E’ stato infine consultato il testo “Corso di sistemi e reti 3” di Paolo Ollari, Zanichelli, Bologna, 2013