Spaghetti Western Distributed Multiplayer Game

Il gioco usa come tema principale il far west, in particolare prende spunto dal film *Il Buono, Il Brutto e Il Cattivo* dal quale derivano le fazioni del gioco:

- CappelliBianchi → Il Buono
- CappelliNeri → Il Cattivo

In base alla fazione scelta, i giocatori vestiranno i panni del Buono o del Cattivo mentre, il ruolo del Brutto sarà affidato al Server.
Lo scopo del gioco è semplice, sconfiggere gli avversari e accumulare più punti possibili (ossia le munizioni) in modo da vincere la partita. Il Brutto, gestito automaticamente dal server, si muove alla fine di ogni turno e disturba gli altri giocatori rubando le munizioni a tutti i giocatori che lo incontreranno in una delle città.
Ad ogni turno, ogni giocatore può scegliere tra 3 mosse possibili:

- spostarsi in un altra città;
- attaccare un avversario;
- passare il turno.

Durante la partita, è possibile trovare le munizioni nelle città della mappa e di raccoglierle al proprio passaggio e di ingaggiare una battaglia per rubare le munizioni ai giocatori della fazione avversaria.
Durante la sfida, il giocatore può scegliere se attaccare, difendersi o ricaricare la propria arma. L'esito della sfida viene deciso dal Server in base alle scelte ricevute dai giocatori.
Durante la partita, le posizioni dei singoli giocatori resteranno nascoste ma, attraverso un segnalino (corrispondente al cappellino della fazione del giocatore) si indicherà la presenza di uno o più giocatori sulla città corrispondente. Solamente spostandosi su una città sarà possibili vedere quanti e quali giocatori si sono fermati li.
La posizione del giocatore è indicata dalla presenza di un omino che rappresenta anche la fazione di appartenenza. A fine partita vincerà una delle due fazioni e verranno anche indicati i giocatori “forti”. Il client, in caso di uscita dalla partita, potrà tentare di collegarsi al gioco anche se già avviato.

Tipologia di comunicazione

Per la comunicazione del gioco è stato deciso di implementare una comunicazione **non** permanente tramite socket, ovvero è stato deciso che la connessione tra Client e Server doveva essere aperta solo per eseguire una specifica *"procedura" e,* dopodiché doveva essere chiusa.
Siccome il server si occupa della propagazione degli sviluppi del gioco, per l'implementazione di questo tipo di comunicazione e per riuscire a inviare gli sviluppi del gioco è stato necessario invertire i ruoli client/server durante le fasi del gioco.
Il gioco si basa sui turni che sono gestiti dal server stesso. Ad ogni turno il server stabilisce una connessione permanente (che dura per tutto il turno) con un Client per notificargli sia l'inizio del turno che per permettergli di giocare il proprio turno. Tutti gli aggiornamenti che il Server invia durante questa fase sono inviati su connessioni differenti.
I dati necessari per contattare il Server, sono fissi e sono integrati nel codice mentre, i dati del client che sono variabili sono resi noti al server durante la prima fase del gioco.
Il protocollo di comunicazione implementato si basa sul protocollo *TCP* il quale ha già integrato una serie di controlli e ritrasmissioni sul corretto invio dei dati.

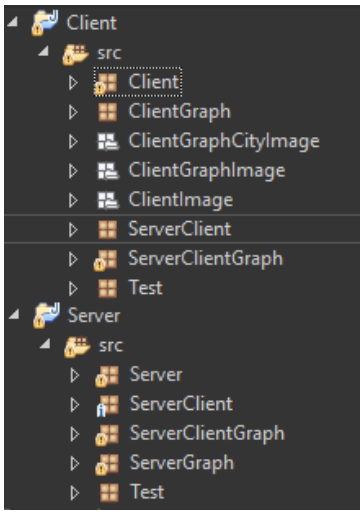
Organizzazione del codice

Poiché si tratta di un gioco online, è stato deciso di sviluppare il gioco dividendolo in due progetti : uno dedicato solo al Client e uno solo per il Server. Questa decisione è stata utile per poter simulare anche in locale "un ambiente online" e testare la serializzazione e il corretto invio delle classi comuni ai due progetti.

Inoltre, per separare i diversi compiti delle classi, i due progetti sono stati ulteriormente suddivisi in vari package; ad esempio il progetto del client è diviso in:

- Client: dove troviamo tutte le classi usate per la gestione del gioco.
- ServeClient: dove troviamo tutte le classi usate per la gestione del gioco inviate dal server.
- ClientGraph: dove troviamo tutte le classi per la gestione del grafo.
- ServerClientGraph: dove troviamo tutte le classi per la gestione del grafo inviate dal server.

Lo stesso ragionamento è stato usate per suddividere le immagini presenti nel medesimo progetto e per organizzare il progetto Server.



Server

Il Server si occupa principalmente del controllo della corretta esecuzione del gioco e di alcuni controlli critici sui dati che i Client non possono eseguire come ad esempio l'unicità del username.
Come detto sopra, il ruolo principale del server è quello di monitorare il corretto andamento delle varie fasi del gioco e di inoltrare le corrette informazioni ai client connessi per permettergli di seguire gli sviluppi del gioco. Per fare ciò sono stati implementati una serie di protocolli per riuscire a gestire in modo indipendente tutte le fasi del gioco.
Fino all'inizio del gioco, il Server vestirà il ruolo del server, una volta iniziata la partita invece, il Server vestirà il ruolo del client e, si occuperà quindi di contattare i vari Client.
Tutte le fasi del gioco sono temporizzate in questo modo riusciamo ad evitare che un Client occupi per un tempo eccessivo una o più fasi del gioco bloccando di conseguenza il gioco stesso.

Fasi principali del gioco (lato server):

1. Inizializzazione dei servizi

Il primo compito del server è quello di inizializzare tutti i servizi e procedure che vengono usati e resi disponibili durante le varie fasi di gioco.
I servizi che il server mette a disposizione dei client sono :

- un servizio di connessione o riconnessione per accedere al gioco;
- un servizio di chat per poter interagire con alri giocatori;
- un servizio di pingPong per dare ai Client la possibilità di controllare lo stato del server stesso.

In questa fase, vengono inizializzate anche tutte le strutture dati indispensabili sia per la comunicazione tra il Server e il Client sia quelle per lo scambio di risorse.

2. Connessione, Registrazione e Riconnessione

In questa fase, i Client possono contattare il Server per o registrarsi ad una nuova partita o per ricollegarsi ad una partita in corso.
Oltre al pool che permette una registrazione concorrente, sono state implementate due procedure di controllo:

- un timer per l'inizio del gioco;
- un pingPong per il controllo dello stato dei Client registrati.

La prima procedura scandisce il tempo di registrazione e permette di anticipare o posticipare l'inizio della partita in base ai giocatori presenti; la seconda, consente di controllare lo stato dei Client connessi e di rimuoverli se necessario. Con l'inizio della partita, le procedure utilizzate vengono chiuse ma il servizio restea disponibile solo ed esclusivamente per dare la possibilità ai Client di riunirsi al gioco.
Prima di riconnettere un Client, attraverso un servizio disponibile sul Client stesso, si verifica il suo stato ed in base a ciò si potrà procedere o meno.

Questa fase coinvolge le seguenti funzioni :

- FasePreliminare();
- InizioFasePreliminare();
- Connessione();
- Riconnessione();
- InizializzazionePartita() (solo per la riconnessione).

3. *Inizializzazione del gioco*

Non appena si da l'inizio al gioco, tutte le informazioni raccolte ed elaborate nella fase precedente vengo inviate e rese disponibili ai Client.
Basandosi sui dati presi, in questa fase, il server crea una mappa di gioco disponendo in maniera randomica le città che la compongono. Anche in caso di errore, questa fase viene eseguita solo una volta dato che i Client possono ricollegarsi e riottenere le informazioni perse prima.
In questa fase, viene creata la mappa di gioco e per fare ciò, viene utilizzata la classe CreatePosGUI. La creazione della mappa, viene basata sul numero dei giocatori presenti e può avere un massimo di 30 nodi (10 giocatori). In questa classe, vengono stabilite tutte le posizioni che dovranno assumere i vertici, e tramite un apposito algoritmo viene controllato che nessun vertice si sovrapponga ad un altro. La creazione dei vertici può richiede più passaggi e, se non si dispone di uno spazio a sufficientemente grande può anche fallire. Una volta settate randomicamente le città, verranno create le strade anch'esse in maniera randomica. Una volta pronti i dati, verranno manditi al client dove la classe Grafox li potrà elaborare.

Questa fase coinvolge le seguenti funzioni :
- Inizializzazione();
- InizializzazionePartita().

4. *Il Gioco*

A questo punto, inizia il gioco vero e proprio e sarà compito del server gestire l'intera partita. I giocatori sono divisi in due liste rappresentanti le due fazioni di appartenenza che vengono usate dal server sia per la gestione dei turni che per la propagazione delle informazioni. Ogni turno, viene giocato da un solo giocatore alla volta e la gestione dei turni permette di alternare i giocatori delle due fazioni.
Ad ogni inizio turno, il server contatta il giocatore interessato che, se disponibile potrà eseguire il proprio turno ed inviare i comandi al server se non disponibile verrà saltato e si passera direttamente al turno successivo. I giocatori che non parteciperanno al proprio turno per un numero stabilito di volte verranno eliminati dal gioco e in questo caso sarà impossibile ricollegarsi.
Una volta iniziato il turno, il giocatore potrà effettuare 3 diverse mosse:
- movimento;
- attacco;
- fineTurno.

Il giocatore non potrà ripetere due volte la stessa mossa ed avrà al massimo 10 secondi circa di tempo per concludere il proprio turno. Allo scadere del tempo massimo a disposizione, viene forzato il fine turno e passerà il turno al prossimo giocatore. La partita dura un tempo settato in base ai partecipanti (compreso tra un minimo di 4 minuti e un massimo di 20).
Per la scansione del tempo della partita è sta utilizzata una procedura apposita che si occupa anche della propagazione dell'avviso di fine partita, della classifica e della chiusura di tutte le risorse utilizzate durante la partita. Oltre alla gestione del turno, in questa fase il server si occupa della propagazione dei risultati dei turni disputati dai vari giocatori. Duranti i vari turni, il server spenderà un po di tempo anche per cambiare la posizione del Brutto in maniera del tutto casuale.

Questa fase coinvolge le seguenti funzioni :
- gioco();
- AggiornamentoComando();

5. Chiusura del server

Questa fase, e la fase finale del gioco ed è composta dalla sola procedura descritta nella fase precedete. Questa procedure si attiva automaticamente alla fine del tempo a disposizione interrompendo la partita in qualsiasi stato si trovi. Per chiudere correttamente il gioco è stato necessario simulare alcune connessioni o fasi di gioco per poter velocizzare o sbloccare le fasi di gioco bloccate.

Servizi attivati dal server:

1. Chat

Questo servizio è reso disponibile dal server durante tutta a fase di gioco. Grazie a questo servizio il Server da l'opportunità ai Client di poter scambiare messaggi sia globali che privati (in base alla fazione). Il compito di questo servizio è quello di analizzare il messaggio ricevuto e di indirizzarlo sulla corretta chat (globale o privata).

2. PingPong (threadPingPong)

Anche questo servizio è reso disponibile per tutta la fase di gioco e consente ai Client di controllare lo stato del Server. Il suo compito è quello di scambiare una brevissima sequenza di messaggi per notificare al Client il proprio stato.

Le procedure usate dal server:

1. Temporizzazione della partita (TimeOut)

Questa procedura viene utilizzata dal server per temporizzare la durata della partita. Il suo compito è quello descritto nella fase di chiusura del server.

2. Temporizzazione della registrazione (TimerRegistrazione)

Questa procedure viene utilizzata dal server per temporizzare la registrazione e far iniziare la partita al momento giusto. Effettua n controlli per controllare quanti giocatori sono presenti e può “decidere” di anticipare o posticipare l'inizio della partita.

3. Controllo Client online (PingPong)

Questa procedure server per poter usufruire del servizio messo a disposizione dal Client per controllare lo stato del Client stesso. Questa procedura server per controllare lo stato dei Client durante la fase di registrazione e, nel caso in cui un Client si disconnetta in questa fase, verrà automaticamente eliminato dalla lista dei giocatori.

4. Controllo munizioni (Munizioni e ThreadMunizioni)

Il compito di questa procedura è quello creare e posizionare sulle città un numero di munizioni in maniera del tutto casuale per renderle disponibili ai giocatori. Il ruolo della procedura “Munizioni” è quello di selezionare la città mentre l'altra procedura “ThreadMunioni” si occuperà di quantificare il valore ed il tempo di visibilità.

Client

Nella prima fase il ruolo principale del Client è quello di raccogliere le informazioni ed inviarle al Server, mentre nella seconda è quello di raccogliere le informazioni inviate dal Server e di elaborarle per aggiornare lo stato del gioco e/o la grafica.
Durante la prima fase di gioco in Client vestirà il ruolo del client mentre successivamente vestirà il ruolo del server.
Come nel server, tutte le fasi del gioco sono temporizzate sia per definire un limite di tempo per cui è possibile usare le risorse del Server che per evitare che resti bloccato sull'attesa di messaggi.
È stato implementato un Client anche su android che differisce solo per l'organizzazione del codice ma mantiene tutte le funzionalita del Client Java a meno della visualizzazione della mappa di gioco.

Fasi principali del gioco (lato server):

1. Inizializzazione dei servizi

Il primo compito del Client è quello di inizializzare tutti i servizi e procedure che vengono usati e resi disponibili durante le varie fasi di gioco.
I servizi che il Client mette a disposizione del Server è :

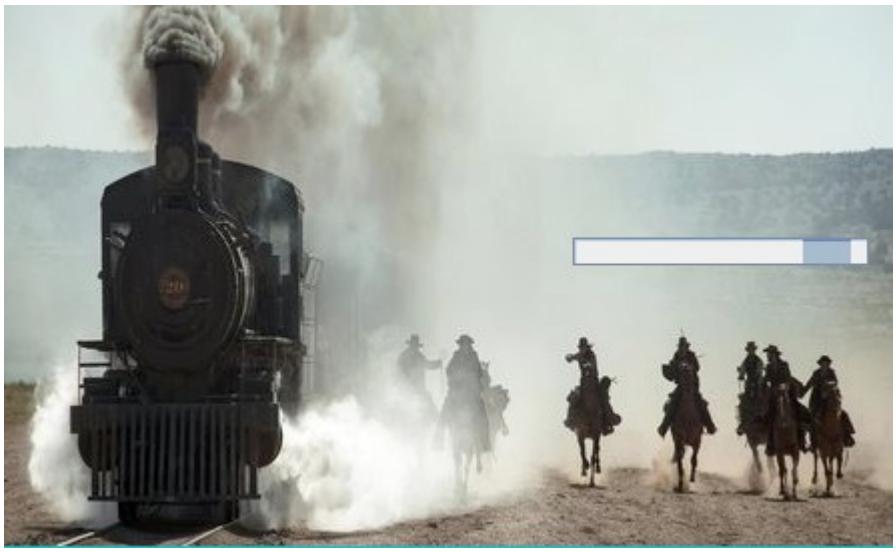
- un servizio di pingPong per dare al Server la possibilità di controllare lo stato del Client stesso.

2. Registrazione e Riconnesione

In questa fase di gioco, viene fornita al giocatore un form dove inserire i dati necessari alla registrazione. I dati inseriti vengono prima analizzati e poi, se corretti inviati. Insieme a questo set di informazioni, il Client invia anche le porte necessarie per essere contattato nelle fasi successive.
Questa fase viene utilizzata anche per il momenti di riconnessione ad una partita in corso; in questo caso il campo della fazione non verrà considerato.
La gestione delle porte, è stata implementata in modo da poter permettere sia l'avvio di più Client sulla stessa macchina che per gestire la riconnessione. Per quest'ultimo caso è stato necessario utilizzare un file d'appoggio per evitare di riutilizzare le porte utilizzate nella sessione precedente. Questo è stato necessario per evitare che il Client ricevesse dei messaggi errati.

3. *Inizializzazione del gioco*

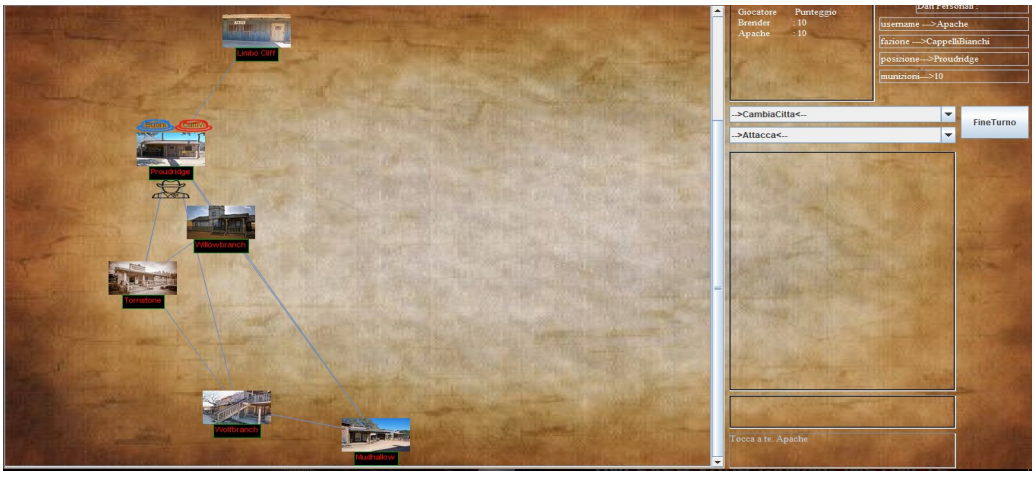
In questa fase, il Client cambia il proprio ruolo e si mette in attesa dell'inizio della parità. In particolare, in questa fase, il Client aspetta che il Server gli invii le strutture vitali per la partecipazione del gioco come ad esempio quella per la creazione della mappa. L'intera fase è controllata da un flag che, se negativo, costringe il Client a riavviare il gioco. Per un'iterazione col giocatore questa fase si basa sulla classe WaitGame la quale carica una finestra di caricamento che si chiuderà a inizio gioco. Per la creazione della mappa di gioco, questa fase si basa sulla classe Grafox la quale, analizzando i dati ricevuti dal server riesce a crearla. Questa classe si occupa esclusivamente della creazione parte grafica; dopo la creazione, la mappa verrà passata alla classe Game per la visualizzazione. Oltre che alla creazione della mappa, che viene fatta una sola volta, un altro compito della classe è quello di aggiornare la grafica della mappa spostando i “segnalini” nelle giuste città.



4. *Gioco*

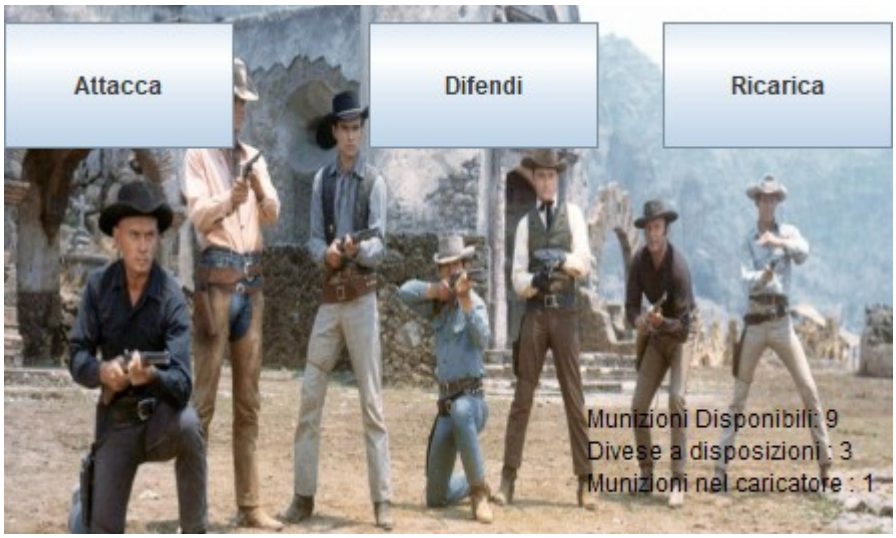
In questa fase, il Client gestisce le tre procedure necessarie per la gestione dellal partita.
Il ruolo principale di questa fase è quello di ricevere le connessioni del Server e di indirizzarle nella procedura corretta. È stato deciso, inoltre di dividere il gioco in tre procedure diverse per poterle eseguire in parallelo. Siccome le connessioni non sono permeanti, ogni connessione comporta l'esecuzione di una delle tre procedure sotto elencate :

- gestione del turno;
- gestione degli aggiornamenti;
- gestione della sfida.



La gestione del turno è affidata alla procedura che permette al giocatore di interagire direttamente col server e di modificare lo stato della partita. La gestione del turno è temporizzata e il giocatore dispone di 10 secondi per concludere il proprio turno.
In questo lasso di tempo potrà spostarsi, attaccare o fare entrambe le mosse. Nel caso di attacco, il turno durerà sino alla fine della sfida.
La gestione della sfida, per lo sfidante, è affidata a questa procedura mente quella dello sfidato è affidata alla procedura di gestione della sfida di conseguenza queste due procedure non potranno mai essere eseguite in parallelo. Per l'esecuzione del turno, questa procedura si appoggia sulla classe Game la quale si occupa sia della gestione della grafica che dell'invio delle mosse, mentre l'analisi dei messaggi inviati dal Server è lasciata alla procedure stessa.

La gestione degli aggiornamenti è affidata alla procedura che può essere attivata in qualsiasi momento della partita e pertanto può essere eseguita in parallelo con entrambe le altre.
Questa si occupa di elaborare e aggiornare lo stato della partita modificando anche la grafica se necessario. I vari aggiornamenti del server sono gestiti in parallelo e viene serializzato l'accesso solo per l'aggiornamento della classifica e il log visualizzato a video nel quale vengono notificati tutti gli aggiornamenti mentre solo alcuni vengono notificati mediante un pop-up. In un aggiornamento particolare, quello di fine partita, questa procedura si appoggia sulla classe CloseGame la quale si occupa della visualizzazione della classifica.



La gestione della sfida è affidata alla procedure che può essere attivata in qualsiasi momento della partita tranne quando il giocatore sta eseguendo il proprio turno, occupandosi solo della sfida da parte dello sfidato.
Durante la sfida, il giocatore avrà 5 secondi di tempo per decidere la mossa, dopo di che verrà generata una automaticamente. La sfida non ha un limite di tempo definiti, tuttavia si risolve in un lasso di tempo ragionevole. Per la gestione della sfida, viene fatto affidamento alla classe Battle che si occupa sia dell'aggiornamento della grafica sia dell'invio dei comandi decisi mentre l'analisi dei messaggi inviati dal Server è lasciata alla procedure stessa.

5. Chiusura del gioco

Questa fase di gioco si occupa della chiusura di tutti i servizi e procedure attivate nelle varie fasi precedenti. Come nel server, per chiudere tutti i servizi è stato necessario simulare alcune connessioni per sboccare alcune fasi bloccate. A differenza del server, in questa fase il Client non chiude completamente il gioco, tale operazione sarà effettuata dal giocatore chiudendo le finestre di gioco rimaste aperte.

I servizi attivati dal Client è:

1. Servizio di PingPong

Anche qui, è presente un servizio per far controllare al Server il proprio stato. Questo servizio viene utilizzato solo in fase di inizializzazione e di riconnessione.
Lo scopo di quest'ultimo è quello di scambiare una serie di messaggi con server per fargli capire lo stato stato.

Le procedure usate dal Client sono:

1. Chat

Questa procedura, serve per poter utilizzare il servizio di chat reso disponibile dal Server. La gestione di questa procedura viene affidata completamente alla classe Game la quale tramite la suddivisione in due sotto procedure riesce a gestire in maniere indipendente la lettura, la scrittura e l'aggiornamento della grafica della chat. Entrambe le procedure durano per tutta la durata della partita.

2. Controllo server online (pingPong)

Questa procedura serve per poter usufruire del servizio messo a disposizione dal server per controllare il suo stato. Questa procedura affianca tutte le fasi del gioco e si attiva allo scadere di un tempo prefissato. Durante la partita è possibile resettare il timer nel caso in cui il Client sia contattato dal Server, nel caso in cui il server non contatti il Client per un lasso di tempo la procedura si attiverà per controllare lo stato del Server.

Progetto a cura di:
Brandi Eugenio
Cruces Hidalgo Nolberto Felix