Relazione progetto Sistemi Distribuiti

Giovanni Pisana*, Giovanni Modica* and Stefano Piscella*

*DISI Università di Bologna, Italia

Contents

Sommario

Il lavoro analizza il processo di sviluppo del gioco "Catch Word", realizzato attraverso il linguaggio di programmazione Java sfruttando le chiamate di procedura remote (RMI). Il prototipo è stato sviluppato per affrontare il problema di tolleranza ai guasti in un ambiente distribuito.

1 Introduzione

Il gioco di seguito descritto consiste in un gioco a tempo dove 2 o più giocatori si sfidano e chi, alla fine dell'ultimo turno raggiunge il punteggio piu alto vince la partita. Il gioco si svolge in n-1 turni dove n sono i giocatori e alla fine di ogni turno viene eliminato il giocatore che ha totalizzato il punteggio minore in quel turno.

Il gioco consiste nell'indovinare il maggior numero di parole nel minor tempo ovvero, i giocatori condividono una plancia di gioco da cui usciranno ogni tot secondi parole generate randomicamente. Se due o più giocatori si trovano ad indovinare la stessa parola sarà uno solo il giocatore che se la aggiudicherà ovvero colui che ha impiegato il minor tempo per scriverla

il sistema è stato progettato in modo tale da essere tollerante ai guasti di tipo crash dei giocatori fino ad un numero di giocatori pari ad n-1, si è imposto che le comunicazioni sono affidabili ossia non si guastano, l'architettura è distribuita e i vari client parlano con un server principale il quale si occuperà della fase iniziale di registrazione dei giocatori e per quanto riguarda le comunicazioni client-server attraverso la rete, sono state implementate utilizzando la tecnologia RMI

2 Il gioco

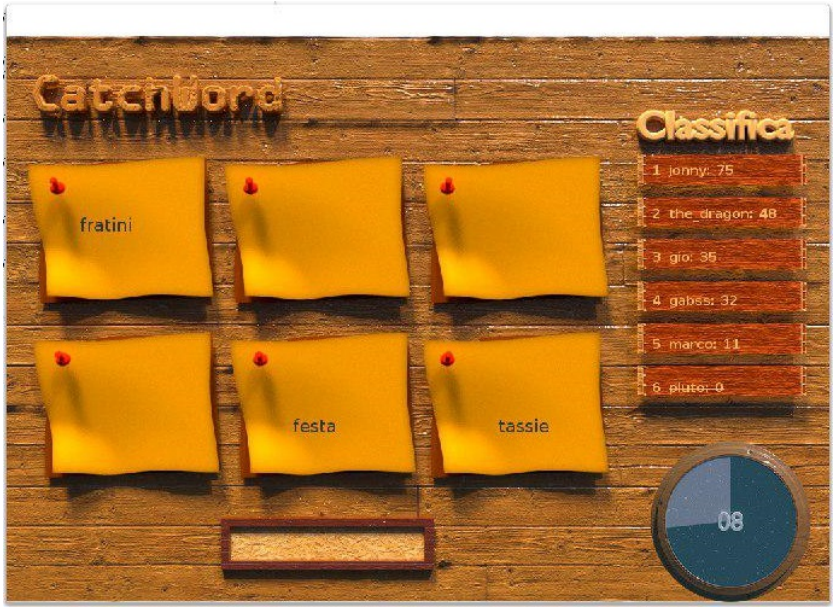

Figure 1: Catch Word

Catch Word è un gioco di abilità multiplayer che permette ai giocatori di sfidarsi sulla loro velocità di scrittura.
La sfida si articola su un numero di turni pari al numero di giocatori meno uno, dove il numero massimo di giocatori è 6. Al termine di ogni turno viene eliminato il giocatore che ha totalizzato il punteggio minore, alla fine dell'ultima manche verrà decretato il vincitore (l'ultimo giocatore rimasto).
Lo scopo del gioco è quello di scrivere il più velocemente possibile le parole che via via appariranno sullo schermo di gioco condiviso. Ad ogni parola "presa", viene assegnato un punteggio in base alla lunghezza della parola e alla velocità con cui viene scritta.
In caso di disconnessione di un giocatore questo viene automaticamente escluso dal gioco e al temine della manche corrente non verrà eliminato nessun altro giocatore.

3 Aspetti progettuali

Abbiamo scelto di attuare una metodologia di sviluppo "Agile" realizzando dapprima un prototipo semplice del gioco al fine di rilevare le principali problematiche corrette poi nella versione finale. Come supporto allo sviluppo abbiamo utilizzato la piattaforma Git - GitLab (https://about.gitlab.com) come sistema di controllo versione del codice, la piattaforma Trello (https://trello.com) per la gestione dei task all'interno del team e si è scelto di utilizzare un IDE di sviluppo unico, IntelliJ IDEA (https://www.jetbrains.com/idea).

3.1 Architettura di sistema

Al termine della prima fase di prototipazione semplice sono stati identificati tre componenti principali che compongono la nostra applicazione ovvero il Network, il Game e la GUI.

Figure 2: Main Component

3.1.1 Network

Il network è il componente che gestisce tutti gli aspetti relativi alla rete. E' in questo componente che vengono affrontati i problemi di fault-tolerance e in cui viene gestito lo scambio di messaggi tra i clients. Si è scelto di isolare il livello di rete in modo da renderlo indipendente dalla logica di gioco e viceversa. Inoltre il Network è l'unico componente che si occuperà delle comunicazioni con il server per quanto riguarda la fase iniziale di registrazione dei clients.

3.1.2 Game

Nel Game verrà implementata tutta la logica del gioco, dalla gestione dei punteggi alla gestione dei round e all'eliminazione dei giocatori. Possiamo identificare questo componente come il nucleo del sistema, che comunicherà, e riceverà comunicazioni, con il livello di rete tramite un adattatore e che esporrà metodi che verranno utilizzato dal componente grafico (GUI).

3.1.3 GUI

La GUI è il componente che gestisce la parte grafica del gioco. Se non fosse per la rilevazione e gestione dell'input da parte dell'utente, che vengono comunicate al Game, rappresenterebbe un elemento totalmente passivo; ovvero il suo compito è quello di richiedere informazioni circa lo stato del gioco al Game.

4 Implementazione

In questa sezione entreremo nei dettagli implementativi dei vari componenti precedentemente analizzati, descrivendo in maniera tecnica le funzionalità del sistema.

4.1 Network

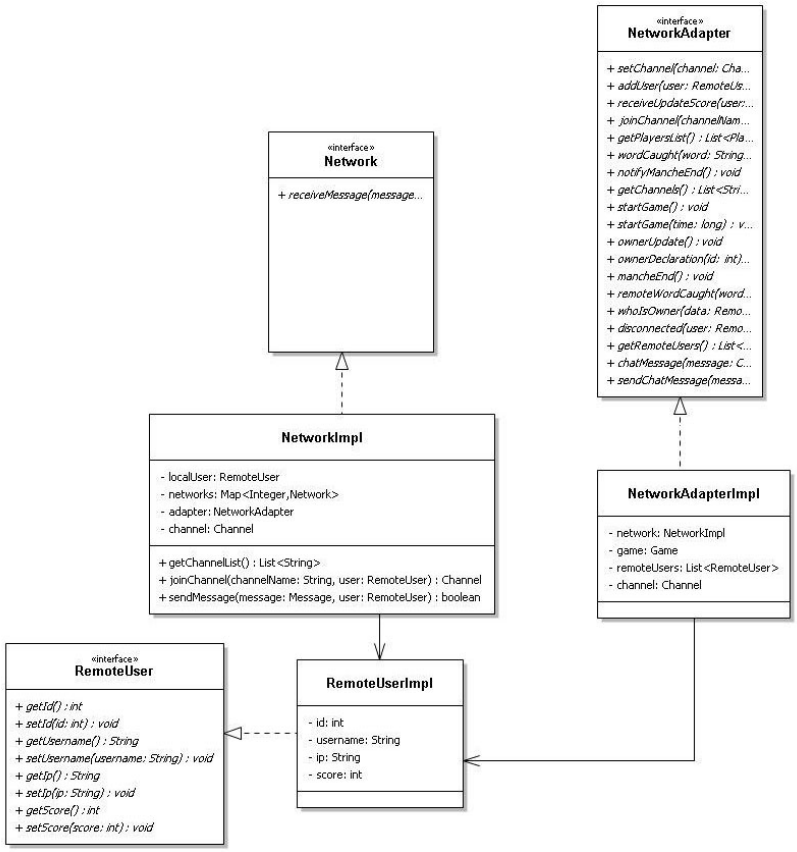

Figure 3: Network

Architettura di rete

Possiamo distinguere due diverse tipologie di architetture:

- Nella fase iniziale di gioco gli utenti devono scegliere su quale canale entrare o se crearne uno nuovo. Queste funzionalità vengono offerte da un server attivo solo nella prima fase di gioco che avrà quindi il compito di tenere una lista dei canali attivi e la relativa lista di giocatori per ognuno di essi. In questo caso, ovviamente parliamo di architettura client-server.

- Tutto il resto del gioco viene gestito in maniera completamente distribuita. Tutti i nodi del sistema sono organizzati in una topologia a mesh completa. E' stata scelta questa tipologia di struttura per garantire l'immediatezza nello scambio dei messaggi, in quanto si tratta di un gioco real-time.

Comunicazioni tra nodi

L'utilizzo di RMI impone che i nodi del sistema comunichino attraverso le chiamate di procedure che risiedono in oggetti cosi detti remoti, ovvero che implementano l'interfaccia Remote. Nel caso in cui queste procedure restituiscano un oggetto, questa chiamata è sincrona, ovvero il client, che ha invocato la procedura, dovrà attendere il completamento di tale prima di continuare con la sua esecuzione.
Quello che per noi risulta essere un problema è il fatto che per ogni tipo di messaggio, che i clients vogliono scambiarsi, è richiesta la creazione di una procedura apposita. Inoltre come per la maggior parte delle comunicazioni distribuite lo scambio di pacchetti dovrebbe avvenire in maniera asincrona.
Per ovviare tali problemi abbiamo scelto di creare un oggetto "Message" che è composto da un tipo, un intero che definisce il contenuto del messaggio e da un contenuto, l'oggetto che si vuole inviare; ed è stata create un unica procedura remota a cui viene passato un oggetto di tipo Message che non ritorna alcun risultato.

Comunicazione con il Game

Per rendere indipendente il livello di rete dalla logica di gioco, è stato introdotto il NetworkAdapter. Il suo compito è quello di nascondere il livello di rete al Game e viceversa. Alla ricezione dei messaggi il Network fa da router instradando il messaggio verso la funzionalità del NetworkAdapter che gestisce quel tipo di pacchetto.
E' in questo livello che vengono implementate le funzionalità di fault-tolerance, di sincronizzazione dei punteggi e della gestione della board condivisa.

4.2 Game

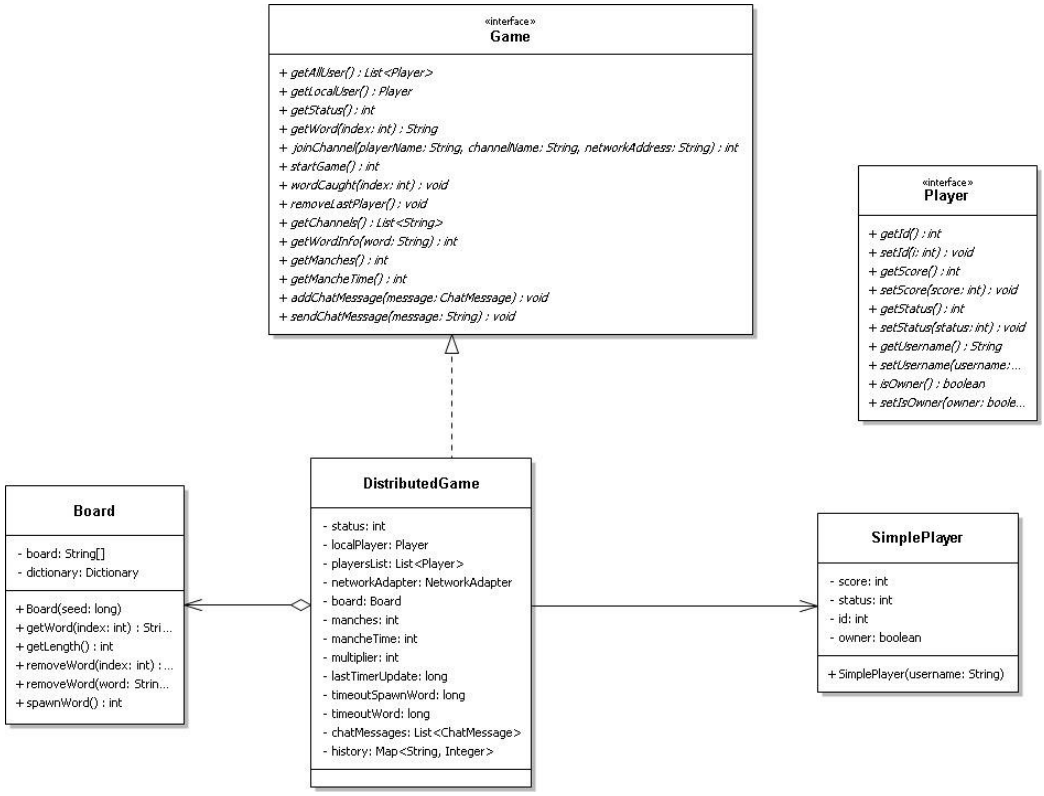

Figure 4: Game

Il Game si occupa di gestire la logica del gioco e per tale scopo si appoggia ad un sotto componente che è la Board. L'interfaccia Game definisce le operazioni utili per il NetworkAdapter, per gestire il collegamento con il Network, e per la GUI. Durante lo sviluppo sono state sviluppate due implementazioni del Game. Dapprima si è sviluppata una versione semplice, SimpleGame, che permette il gioco solo in modalità locale. Successivamente si è sviluppata la versione finale, DistributedGame, che aggiunge le funzionalità di rete.

4.2.1 Board

L'oggetto Board rappresenta la plancia di gioco ovvero contiene il riferimento alle parole attualmente visibili nel gioco e offre funzionalità per la rimozione di queste e la generazione random di nuove.
Per garantire che tutti i clients condividano le stesse parole allo stesso istante vengono utilizzati due oggetti condivisi:

- Un dizionario rappresentato da un file di testo contenente l'insieme di tutte le possibili parole. Il file di testo viene incluso nel gioco.
- Il seme del generatore random. All'atto della creazione di un canale, viene associato ad esso un ID numerico che rappresenterà questo seme. Grazie ad esso i clients che sono all'interno dello stesso canale (condividono lo stesso seed) vedranno comparire le stesse parole nello stesso ordine

4.3 GUI

La GUI offre tre pannelli:

- Un pannello iniziale (StartPane) che permette all'utente di accedere o di creare un canale
- il pannello successivo (ChannelPanel) che rappresenta la lobby di gioco. I giocatori possono chattare tra di loro mentre attendono l'inizio del gioco.
- Infine il pannello principale (GamePanel) rappresenta la plancia del gioco dove i giocatori potranno interagire con il sistema.

4.4 GameEngine

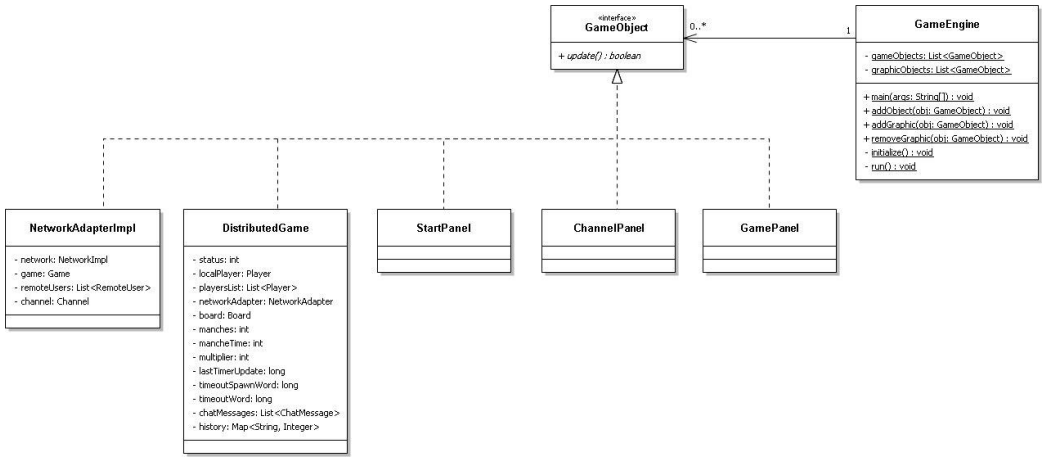

Figure 5: GameEngine

In tutti i giochi esiste il concetto di frame rate legato alla frequenza di aggiornamento dello schermo. Possiamo considerare questi frame come un arco di tempo richiamato ciclicamente nel quale vengono effettuate determinate operazioni (esempio: nella GUI ad ogni frame aggiorniamo la classifica).
Per implementare questo comportamento, indispensabile nei giochi real time, abbiamo valutato una serie di framework java esistenti. Abbiamo però deciso di non utilizzare nessuno di questi poiché la maggior parte rappresentava librerie monolitiche che includevano tanti altri concetti oltre il frame rate (grafica, fisica, etc..).
Abbiamo quindi deciso di sviluppare una semplice soluzione interna, creando il nostro motore di gioco "GameEngine". Il comportamento principale di questo oggetto è un ciclo continuo con pause di tempo fissato (frame rate), all'interno del quale vengono invocati tutti gli oggetti che si registrano al GameEngine (GameObject)

5 Conclusioni

Catch Word è un gioco totalmente originale, non esiste infatti nessun gioco con le stesse caratteristiche. La soluzione è stata sviluppata completamente dal team senza l'uso di framework esterni. Si è cercato di ottimizzare le caratteristiche di RMI garantendo una struttura semplice e facilmente modificabile.
Riguardo gli aspetti di rete, il gioco da funzionare perfettamente nella struttura "laboratorio Ercolani". Sono state affrontate problematiche relative alla sincronizzazione dei punteggi, della plancia condivisa, della lista dei giocatori e della gestione dei crash dei processi fino a N-1 (N numero di processi).
Non è stata considerata la situazione in cui le connessione possano essere inaffidabili, ovvero non consideriamo la possibilità di inviare di nuovo un pacchetto precedentemente spedito poichè lo consideriamo sicuramente consegnato. Inoltre il gioco si basa sulla supposizione che tutti i processi abbiano un ottima connessione, infatti in mancanza di risposta entro un lasso di tempo il processo viene eliminato. In caso di pubblicazione del gioco questi due aspetti sono da valutare attentamente e un possibile futuro sviluppo si orienterebbe di sicuro su questo ambito.