1. INTRODUZIONE

Il gioco sviluppato è un'*Escape Room* in cui è possibile scegliere uno specifico scenario e il tipo di giocatore, principale o secondario. In questa versione è stato implementato un solo scenario, *Prison Break*, con un codice strutturato per facilitare l'aggiunta di nuove ambientazioni.

2. IMPLEMENTAZIONE

La comunicazione tra *client* e *server* è asincrona e avviene tramite il protocollo TCP perché è necessaria l'affidabilità nella trasmissione dei messaggi. Il formato dei dati inviati usa il protocollo *text* perché vengono scambiate solamente sequenze di caratteri. In questo modo non è necessario che il client sia a conoscenza delle strutture dati usate dal *server*, favorendone anche una gestione più snella seguendo il modello del *thin-client*. I *socket* utilizzati sono di tipo bloccante, ad ogni azione del client corrisponde sempre una risposta da parte del server.
Server e client utilizzano entrambi un *buffer* da 1024 byte, ad ogni *send* trasmettono l'intero contenuto del buffer. Ho deciso di non implementare un ulteriore send per l'invio della lunghezza del messaggio perché con le connessioni attuali inviare un pacchetto da 1024 byte non giustifica una doppia send, nemmeno in caso di perdita e rinvio del pacchetto.

2.1 SERVER

Il server è di tipo iterativo basato su I/O Multiplexing, può gestire una richiesta alla volta mentre le altre vengono messe in coda. Il server è in grado di reggere più sessioni simultaneamente quindi è possibile avviare più partite in parallelo, con una coda di richieste di 20 giocatori collegati contemporaneamente. Non è stato impostato un limite di partite che può gestire vista la leggerezza del gioco implementato, con uno più pesante sarebbe stato necessario limitare il numero di partite in contemporanea per non rovinare l'esperienza di gioco agli utenti.

2.1.1 STRUTTURE DATI

Sono state implementate diverse strutture dati per facilitare la gestione del gioco:

- ***user*** salva i dati relativi ad un utente;
- ***session*** mantiene i dati di una sessione attiva;
- ***set*** rappresenta lo scenario;
- ***location*** usata per rappresentare i luoghi esplorabili di uno scenario;
- ***object*** utilizzata per indicare gli oggetti utilizzabili nello scenario;
- ***riddle*** rappresenta i quiz, con domanda e la relativa soluzione;

2.2 CLIENT

Il client è stato implementato il più leggero possibile così da poter girare su qualsiasi macchina. Tutti i comandi inseriti vengono trattati come messaggi indipendenti che vengono elaborati dal *server* che ne fornirà una risposta, questo per seguire il modello *thin-client* e perché nella maggior parte dei casi il *server* è più performante rispetto ad un *client*. La gestione delle risposte da parte del *client* non è subordinata

ad una *listen* dopo la *send* in quanto la gestione dell'interazione con il secondo *client* avviene in maniera asincrona per cui sarebbe stata una ripetizione inutile dello stesso codice. Inoltre, questo migliora leggermente la velocità di elaborazione del *client* e può essere utile per future implementazioni di features che prevedono l'invio di messaggi direttamente dal *server* al *client*.

2.3 FUNZIONALITÀ A PIACERE

La funzionalità aggiuntiva prevede l'inserimento di un secondo giocatore nella partita. Il ruolo del secondo giocatore è di *secondino corrotto*, il suo compito è aiutare a far evadere il prigioniero dello scenario *Prison Break*. Una volta eseguito il login, il terminale del giocatore secondario viene "bloccato", qualsiasi comando inserito non viene inoltrato al server, almeno che non venga immesso il comando *end*, che prevede la chiusura anticipata dalla sessione di gioco. Il terminale si sblocca quando il giocatore secondario viene interpellato tramite l'oggetto *telefono* dal giocatore principale. In quel momento gli viene chiesto quante monete vuole richiedere al giocatore principale per farlo evadere. Se il giocatore principale detiene abbastanza monete il gioco è finito, altrimenti il giocatore secondario torna in standby e il giocatore primario può continuare ad esplorare lo scenario in cerca di altre monete o di terminare la partita per conto proprio. Per la realizzazione sono stati aggiunti token secondari nell'ambientazione.

2.5 IPOTESI AGGIUNTIVA

Per una gestione più rapida e graficamente migliore ho deciso di ridefinire il formato del comando *start* includendo oltre allo scenario anche la tipologia di giocatore all'interno della partita. In questo modo con un solo comando possiamo entrare direttamente in gioco.

***start** codice tipo*

3. SOLUZIONE

Per lo scenario implementato ci sono due vie per vincere, una soluzione principale ed una secondaria che necessita però del collegamento del *client* aggiuntivo (giocatore secondario).
Per la soluzione principale è necessario raccogliere il telefono (sotto il letto) e utilizzarlo insieme al sapone (dentro la scatola) per creare l'oggetto bomba. Successivamente utilizzare la bomba insieme alle sbarre per fare un buco nella prigione e scappare. Nella soluzione secondaria abbiamo il secondo giocatore che rappresenta un secondino corrotto che, con la giusta quantità di monete, ti fa scappare dalla prigione. Quindi è necessario raccogliere un numero x di monete (fino a 3) e sperare che il secondino ci chieda un numero minore o uguale al numero di monete che abbiamo raccolto.

3.1 SOLUZIONE DEI QUIZ

- La soluzione al quesito del tempo di andata e ritorno è: Anna.
- La soluzione al quesito del della sequenza numerica è: 312211