

Conquerors of mars

Federico Bonetti

Sommario

- Decrizione del gioco
- Requisiti principali
- Architettura Client-Server centralizzata
- Pattern utilizzati
- Uno sguardo al design
- Test

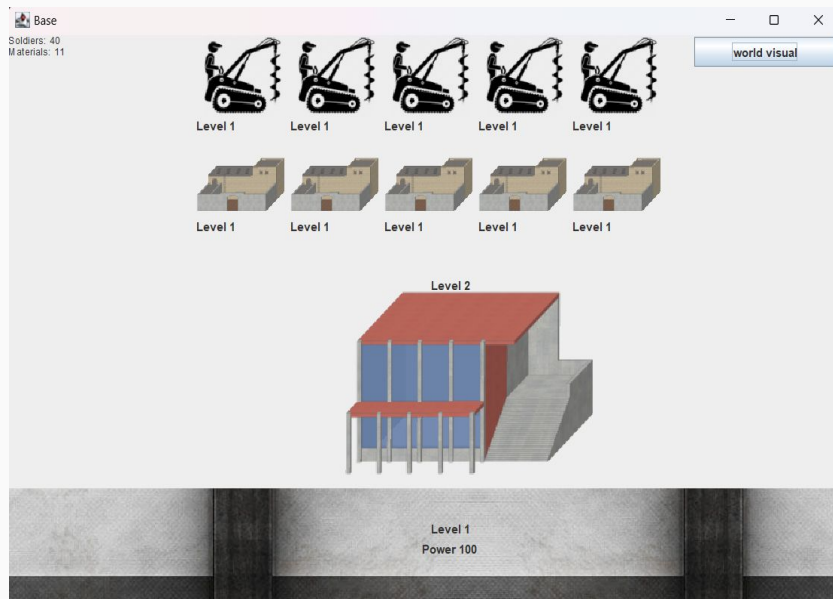
Descrizione del gioco

Si tratta di un gioco di strategia a tempo reale. L'utente deve gestire le risorse della propria base nel tempo, per acquisire più beni possibili e potersi difendere da eventuali attacchi esterni.

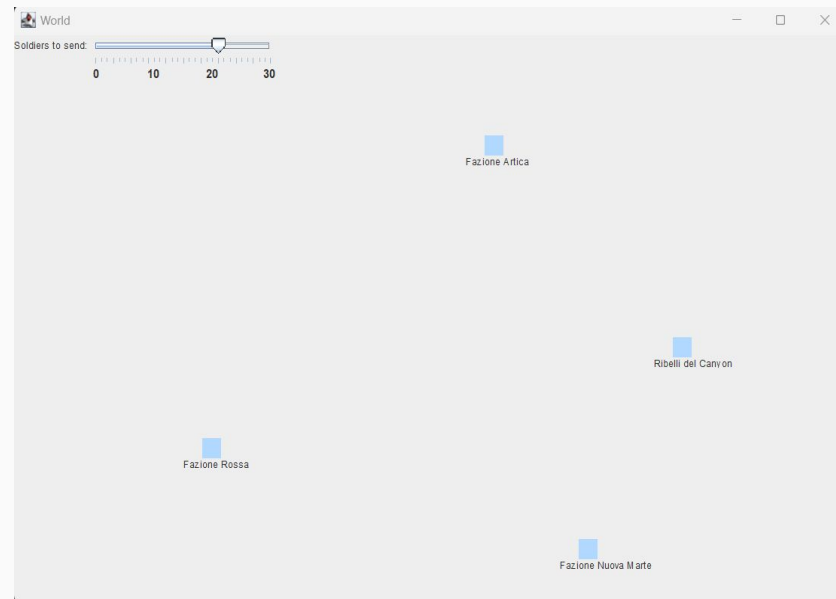
Il gioco si divide principalmente in 2 fasi:

- Gestione della base (gestisci i componenti per incrementare al massimo il guadagno delle risorse e rendere la base più resistente da attacchi)
- Gestione delle truppe (il giocatore può decidere di impegnare i propri soldati in attacchi, con l'obiettivo di rubare risorse a basi nemiche)

Descrizione del gioco

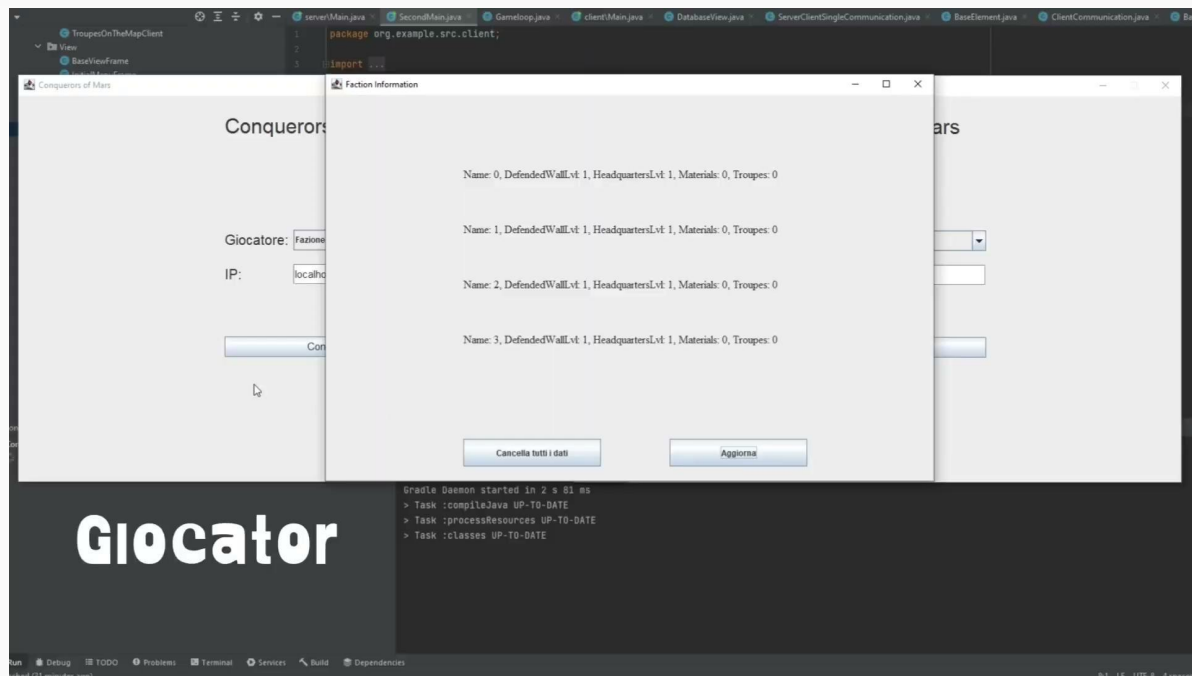


Visuale della base



Visuale del mondo

Descrizione del gioco



Requisiti principali

Funzionali:

1. Lo stato degli elementi deve essere aggiornato nel tempo (anche se i giocatori non sono online).
2. Il giocatore deve poter acquistare nuovi Material Collector o Military Base e incrementarne l'efficienza facendoli salire di livello.
3. Il giocatore deve poter attaccare basi nemiche con l'obiettivo di rubare materiali agli avversari.

Non funzionali:

1. Il software è pensato per funzionare nel lungo periodo.

Architettura Client-Server centralizzata

Visto che tra i requisiti vi è la necessità che il gioco continui ad aggiornare lo stato delle risorse anche in assenza di giocatori online, si rende necessario:

- Architettura client-server: il server gestisce la logica del gioco e manda messaggi ai client per aggiornarli sullo stato.
- Sistema centralizzato: il server è l'unico punto di controllo del gioco e continuerà a funzionare anche in assenza di giocatori online.

Architettura Client-Server centralizzata

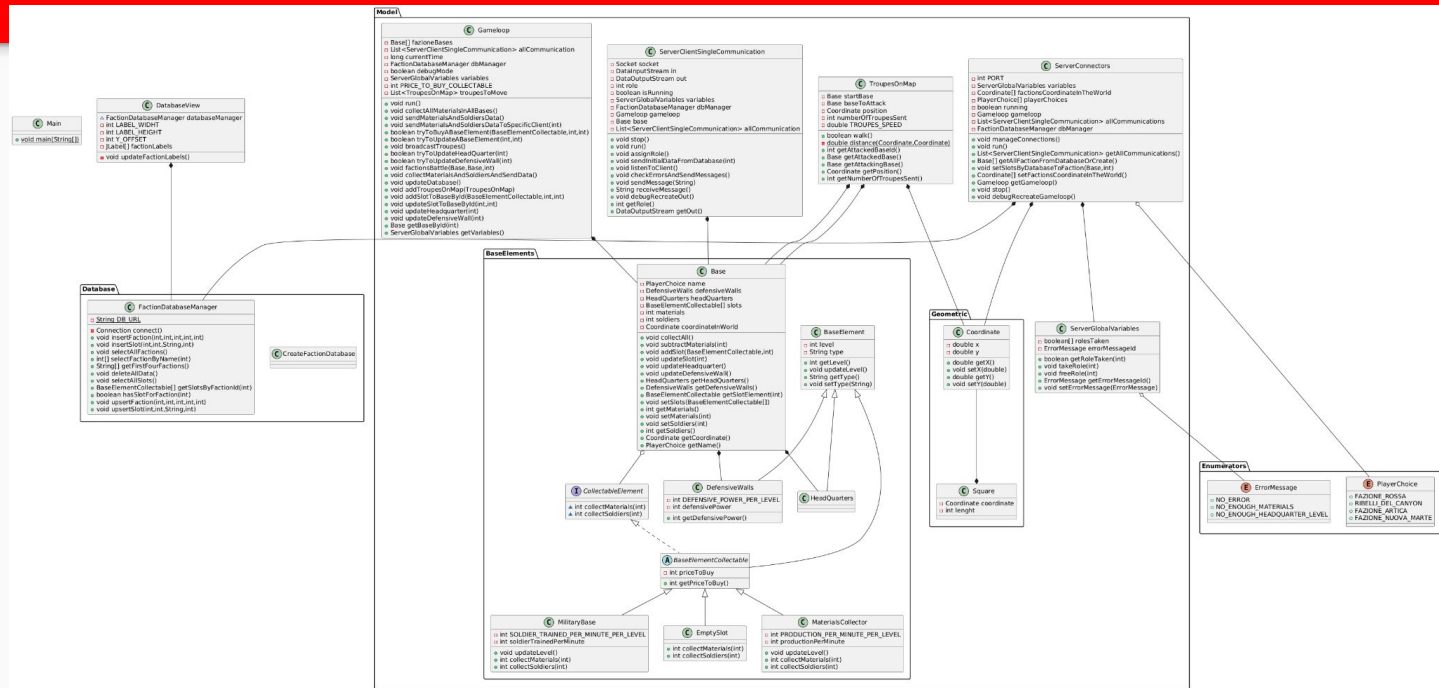
Tecnologie usate:

- Socket: perchè consentono una comunicazione bidirezionale in tempo reale affidabile e senza perdite (protocollo TCP).
- SQLite database: il server salva i dati in un database ad intervalli regolari per permettere in caso di crash del sistema di minimizzare i danni.

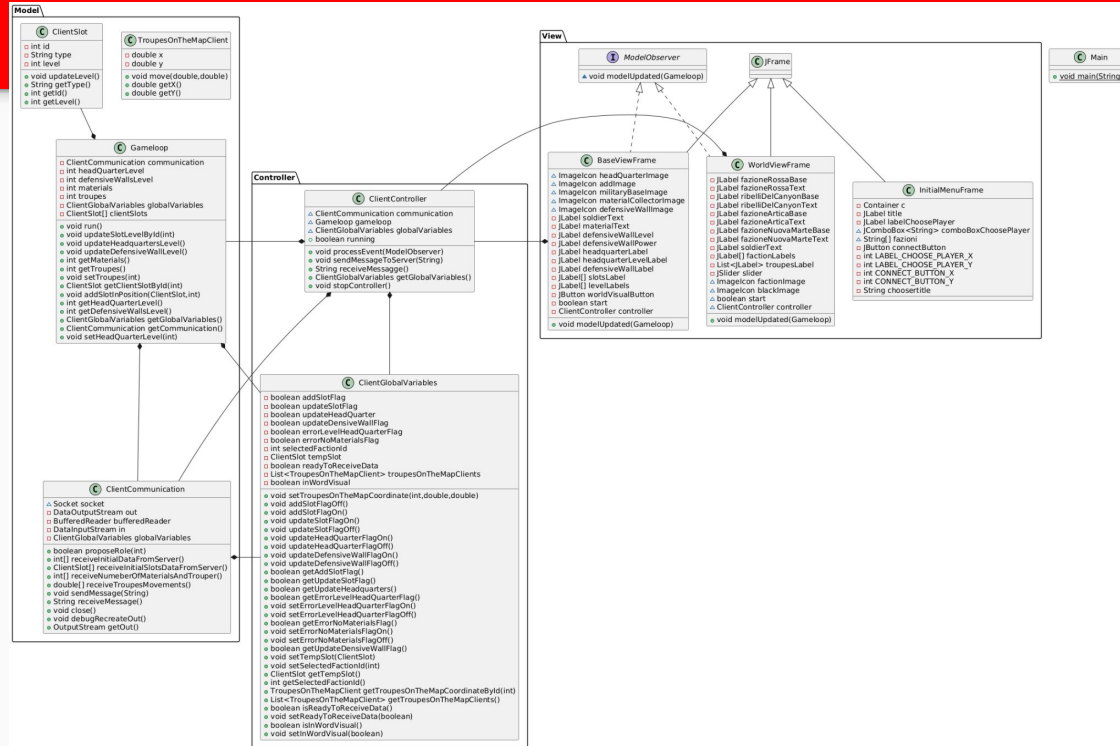
Pattern utilizzati

- Event-Driven pattern: per gestire eventi come il movimento delle truppe.
- Request-Response pattern: per gestire azioni da parte del client che necessitano di una conferma dal server, come l'acquisto di un elemento o la richiesta di poterlo potenziare.
- Time-Based pattern: per gestire l'ottenimento di materiali o l'addestramento delle truppe. Il server aggiorna i client ogni minuto a riguardo.

Uno sguardo al Design (Server)



Uno sguardo al Design (Client)



Come è stato testato il gioco?

- Test di Junit delle principali funzionalità
- Gameplay lungo (2 ore) da 2 computer diversi per testare la longevità del software