IRC Server

Tommaso De Nicola 2006686

Gennaio 2024

1 Introduzione

Il sistema preso in considerazone è un server IRC (Internet Relay Chat), implementato in C++ e provvisto anche con un database gestito tramite PostgreSQL per gestire i canali e i log. Gli utenti interagiscono con il server utilizzando il loro client IRC preferito (in questo caso utilizzeremo KVIrc), dove si potranno utilizzare i classici comandi IRC per entrare/uscire/chattare in vari canali ed altro ancora.

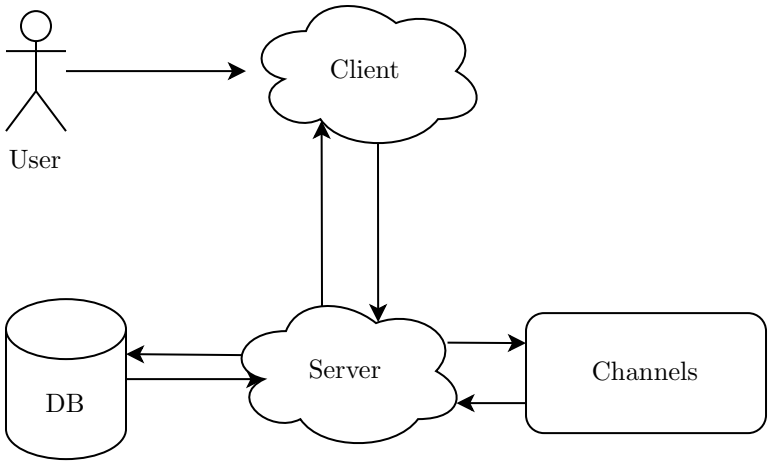

Figure 1: Ambiente

2 Requisiti Utente

Questa è una lista dei comandi che possono essere utilizzati all'interno del server per rendere più facile la comunicazione e l'utilizzo del server stesso.

Non Registered User

1. Nick: il nickname per la registrazione
2. Pass: la password del server
3. Quit: esce dal server
4. User: lo username per la registrazione

User

1. Join: entra in un canale (password opzionale)
2. Notice: manda un messaggio ad un solo utente in un canale
3. Part: esce dal canale
4. Ping: invia un controllo di connessione
5. Pong: ritorno del controllo di connessione
6. PrivMsg: apre una chat privata con un altro utente

Admin

1. Kick: caccia dal canale

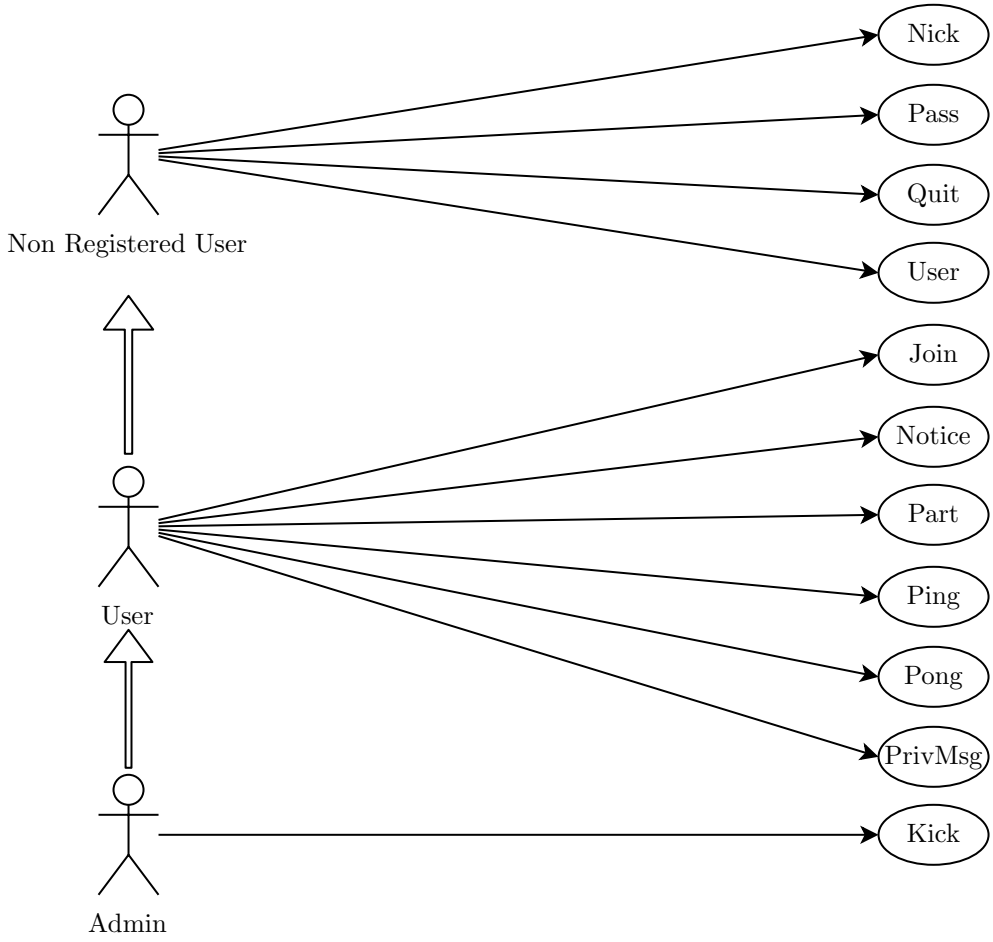

Figure 2: Use Case

3 Requisiti di Sistema

I requisiti di sistema che necessitiamo sono, partendo dal componente principale al quale si connetteranno gli altri componenti è il "Server", questo sarà connesso agli altri componenti come ad esempio il modulo di comunicazione con il database, o il modulo per i canali che potrà caricare canali precedentemente creati direttamente dal database, e gestire tutte le azioni correlate ai canali, poi abbiamo il componente "Client" che ci permette di rappresentare i client e utenti connessi al nostro sistema, che chiaramente potranno usufruire dei vari comandi tramite il gestore apposito.

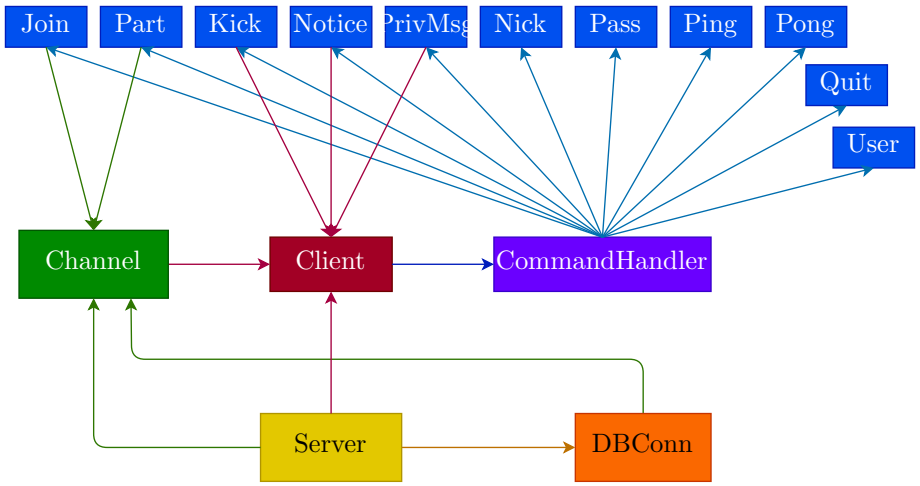

Figure 3: Componenti

Ecco un esempio di un messaggio privato tra due utenti, e come il server con i suoi componenti lo gestiscono.

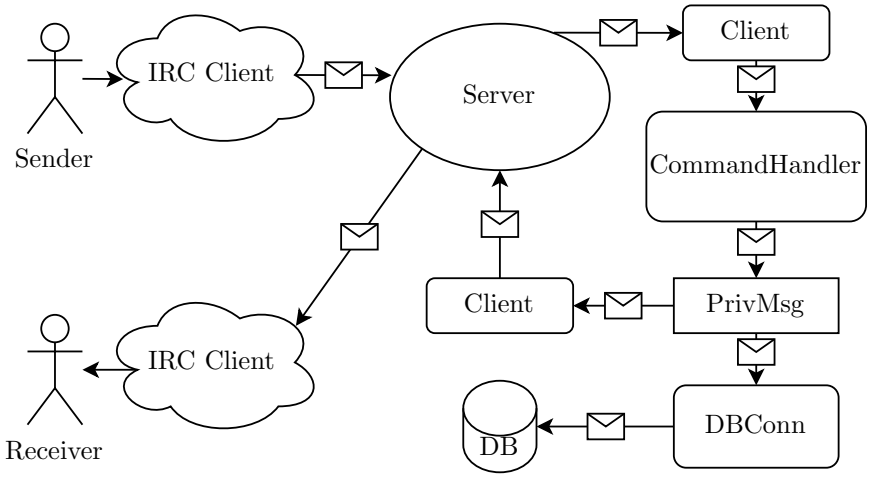

Figure 4: Esempio di Invio di un Messaggio Privato

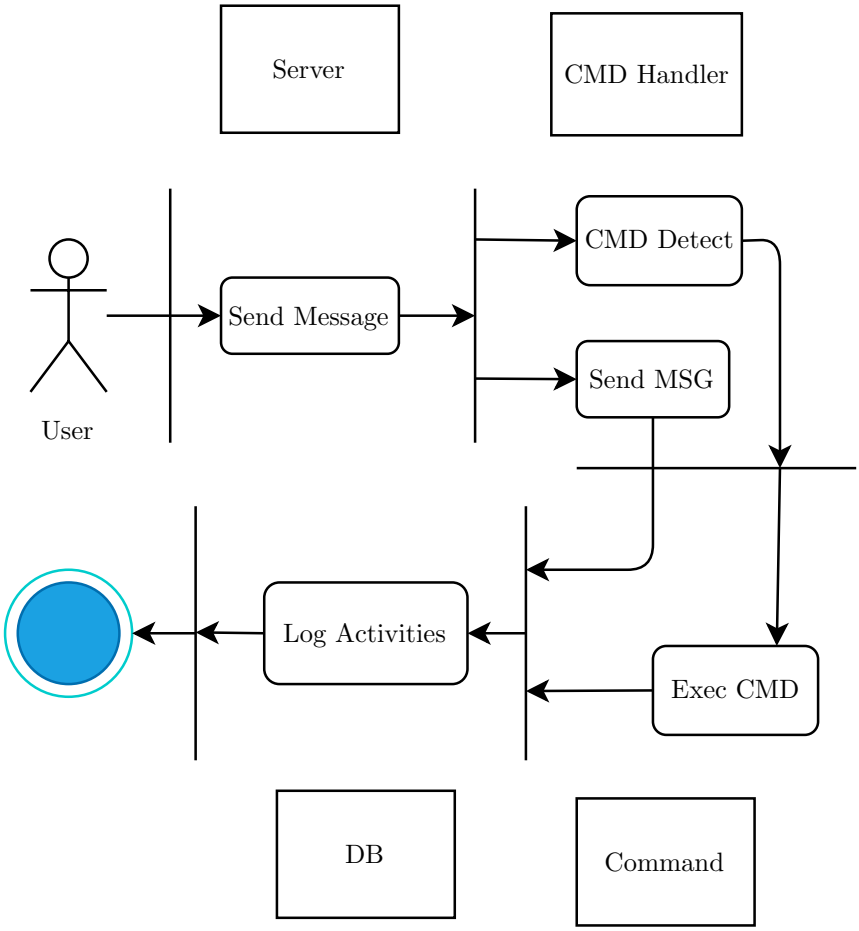

Figure 5: Activity Diagram

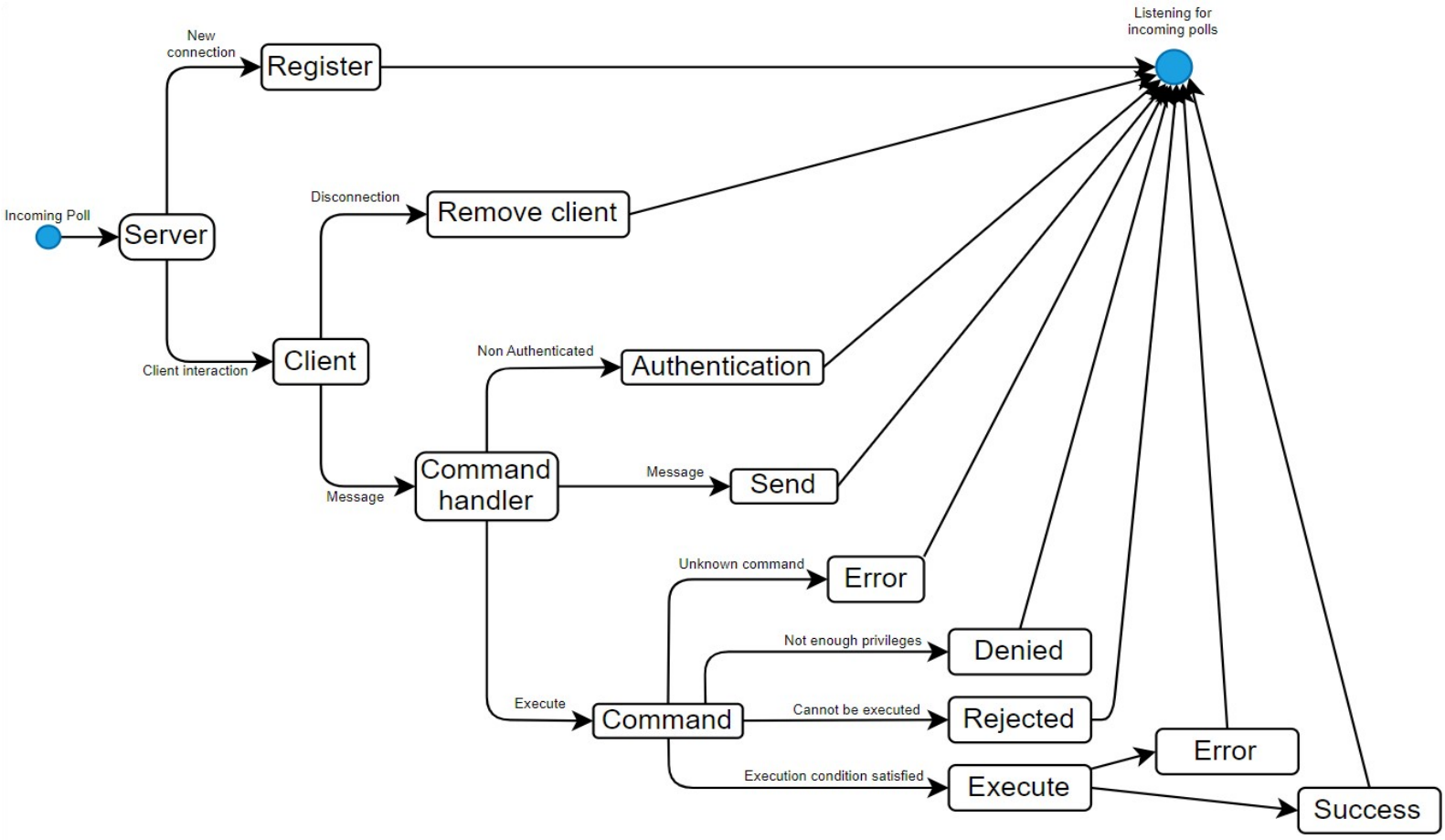

Figure 6: State Diagram

4 Implementazione

4.1 Componenti

4.1.1 Server

Il componente del server, quando vene creato, carica dal database i canali creati e salvati precedentemente, dopodichè apre il socket principale in ascolto di richieste di connessione, tutte le connessioni verranno gestite tramite dei polling events che chiameranno i vari gestori di connessione, disconnessione e messaggi; nel gestore dei messaggi verrà poi chiamato il gestore dei comandi per poi evntualmente eseguirne.

4.1.2 Client

Quando un un poll envent di connessione viene innescato, un nuovo client viene creato con il suo socket e identificativi; esso può messaggiare, entrare, creare, uscire anche da canali.

4.1.3 Channel

Un canale può essere creato da un utente o caricato da database, esso può possedere una password opzionale per ragioni di sicurezza; un normale messaggio in un canale viene inviato ad ogni altro utente presente in quel canale, può anche aggiungere, rimuovere oppure anche cacciare utenti (quest'ultimo solo per admin); in automatico l'admin di un canale è il creatore stesso.

4.1.4 CommandHandler

Esso Contiene tuttle le istanze dei vari comandi e le gestisce, quando un messaggio viene ricevuto, esso viene analizzato per identificarlo come un normale messaggio, un comando inesistente, o un comando valido (per essere eseguiti il client deve avere un sufficiente livello d'autorizzazzione).

4.1.5 Command

Il componente "Command" è una astrazione utilizzata come interfaccia per ogni comando per poter definire delle necessità di base,

ognuno avrà quindi un livello di atorizzazzione richiesto per essere invocato, e la funzione d'esecuzione del comando in se; ogni comando ha anche delle condizioni che devono essere verificae prima della sua invocazione.

4.1.6 DBConn

On creation it connects to the Database, it can execute commands, like insert, delete or update, or can execute queries, it also provide a logging method to monitor the server activities that are also saved on the database. Alla creazione si connetterà al database, esso può eseguire comandi, come inserimento, cancellazzione o aggiornamento dal database, o può eseguire delle interrogazioni al database, esso inoltre fornisce anche un sistema di logging per monitorare il sistema e ne archivia una copia nel database.

4.2 Database

Lo schema del database è composto da 2 tabelle, la prima per i logs, che hanno un id unico utilizzato per differenziarli tra di loro, i log in se, e la "data", cioè il timestamp del momento dell'evento segnato; la seconda tabella è dedcata ai canali, essa è utilizzata per salvare i canali con le loro informazioni, quali nome e password opzionale, questi verranno caricati all'avvio del server, quando invece viene creato un nuovo canale, esso viene inserito all'interno della tabella automaticamente.

```
CREATE TABLE logs (
  id SERIAL PRIMARY KEY,
  log VARCHAR(1023),
  date TIMESTAMP
);

CREATE TABLE channels (
  name VARCHAR(255) PRIMARY KEY,
  password VARCHAR(255) NULL
);
```

Listing 1: DB Schema

5 Risultati

Questo server offre una normale esperienza IRC, con tutte le dovute comodità; invece di essere solamente sperimentale, questo progetto è effettivamente utilizzabile come server (si consiglia KVIrc come client per la facità d'utilizzo).