Relazione progetto Advanced Programming Languages

Marchese Angelo (O55000395)

Link Repository:

https://github.com/ilconte96/bitTorrent.git

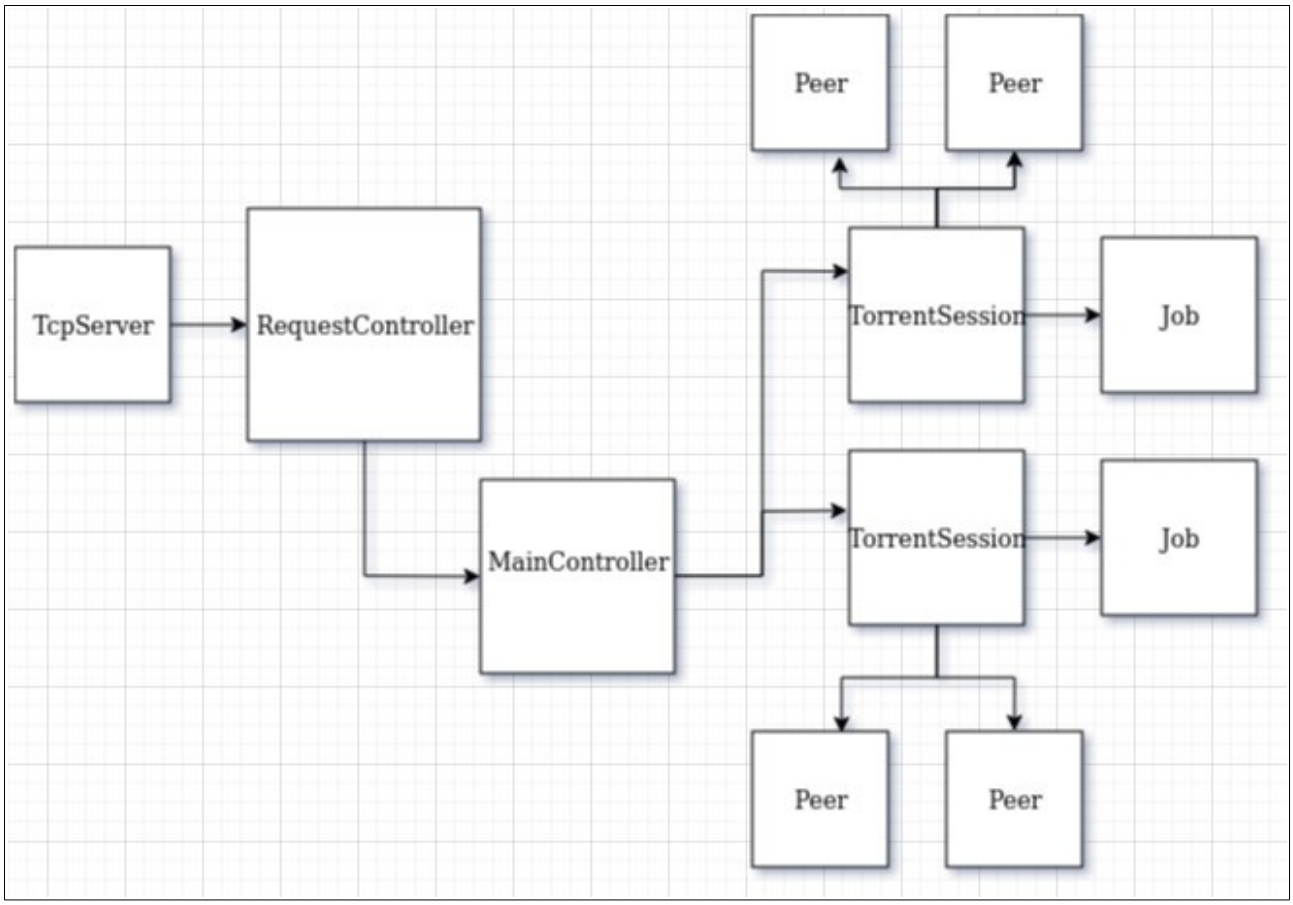
SOMMARIO

1 INTRODUZIONE

E’ stato realizzato un ambiente simulativo per un sistema peer-to-peer basato sul protocollo BitTorrent. In particolare sono stati implementati un tracker ed un client che per la maggior parte degli aspetti sono conformi al protocollo. Il progetto è disponibile al repository: https://github.com/ilconte96/bitTorrent.git dove vengono pure indicati i vari passi per avviare sia il tracker che il client.

2 TRACKER

Il tracker è stato realizzato utilizzando il linguaggio C++, con l’idea di adottare un paradigma di modellazione dei componenti object-oriented. Di seguito viene mostrato uno schema dell’organizzazione dei vari componenti del programma:

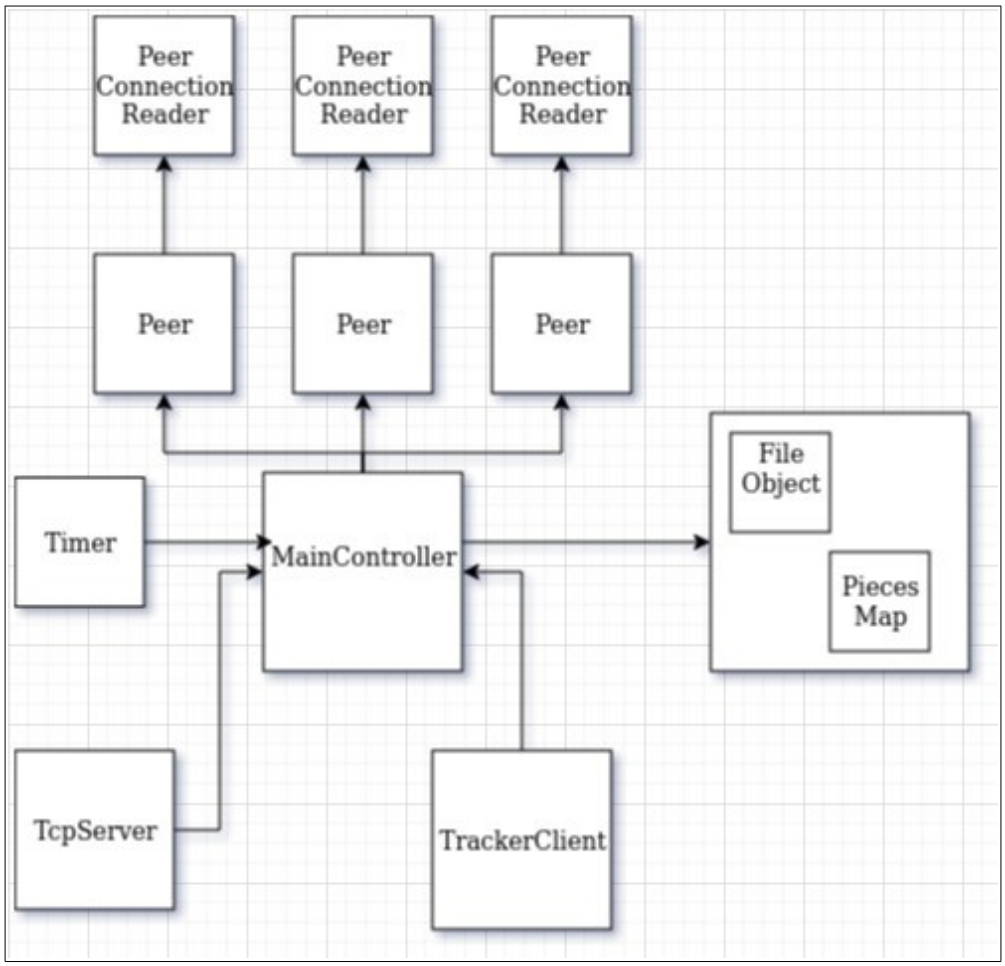


Il TcpServer sta in ascolto delle richieste provenienti dai vari peer. Per ciascuna richiesta viene instanziato un nuovo thread che richiama il metodo serveRequest del RequestController per gestire la richiesta. In particolare quest'ultimo componente sulla base del tipo di richiesta chiamerà uno specifico metodo del componente MainController e alla fine restituirà una risposta. Il MainController si occupa della gestione delle varie TorrentSession e dunque delle relative operazioni di creazione ed eliminazione. Poichè possono esserci accessi concorrenti da parte di più thread alla struttura contenente le TorrentSession, questa è protetta tramite meccanismi di sincronizzazione. Ogni TorrentSession gestisce l'insieme dei Peer che ne fanno parte ed anche in questo caso l'accesso alla struttura contenente i vari Peer è protetta da meccanismi di sincronizzazione. Anche se le operazioni di aggiunta ed eliminazione di TorrentSession vengono sequenzializzate quelle che richiamano metodi di due TorrentSession differenti possono avvenire parallelamente. Per ogni TorrentSession vi sarà un Job che periodicamente eliminerà i Peer per i quali non è stato ricevuto un messaggio di heartbeat.

3 CLIENT

Il client BitTorrent è stato realizzato utilizzando il linguaggio Python. Anche in questo caso la scelta è stata quella di orientarsi verso il paradigma object-oriented, ma in aggiunta a questo è stato utilizzato il modello di programmazione ad attori con l'obiettivo di sfruttare a pieno il grado di parallelismo ed asincronia offerto da quest'ultimo. A supporto della programmazione con il modello ad attori è stata utilizzata la libreria pykka (www.pykka.org) che si ispira alla libreria Akka per Scala.

Di seguito è riportato uno schema dell'organizzazione dei componenti:



Il TcpServer gestisce le richieste di connessione da parte degli altri peer, mentre il TrackerClient contatta periodicamente il tracker inviando un messaggio di heartbeat ed ottenendo una lista dei peer presenti nella torrent session. Il nucleo dell'applicazione è rappresentato dal MainController, l'attore principale che riceve messaggi dall'esterno e sulla base di questi prende decisioni ed invia messaggi agli altri attori secondari, i Peer. Ciascun attore peer modella la connessione con un altro neighbour peer, effettuando richieste per i chunks e rispondendo ad eventuali richieste da parte del neighbour peer. Per ogni connessione con un neighbour peer vi è un PeerConnectionReader che sta in ascolto di eventuali messaggi ricevuti. Le strutture PiecesMap e FileObject vengono utilizzate per gestire la concorrenza nell'accesso al file e alla mappa dei chunks. Il Timer periodicamente invia un messaggio al MainController per avviare l'algoritmo di choking.