Relazione

Livello trasporto

Ho scelto il protocollo TCP perché per un’applicazione di messaggistica è preferibile l’affidabilità al throughput

Livello applicazione

Ogni pacchetto è composto da un header e da un payload

|  |  |
| --- | --- |
| 1B | 4B |
| Tipo del messaggio | Lunghezza del payload |

* Header:
* Payload: cambia struttura a seconda del tipo del messaggio, questa è documentata nel dettaglio in *global.d/network\_tools/network\_tools.h*

Gestione delle richieste lato server

Il server gestisce le richieste in modo **sequenziale** con *select*, ho fatto questa scelta per evitare la complessità che deriva dalla programmazione concorrente e per facilitare la condivisione delle strutture dati comuni a tutte le connessioni.

Per evitare che il server rimanga in idle attendendo un intero messaggio da un device lento, il contenuto del messaggio viene bufferizzato in una struttura globale e utilizzato solo quando si riceve l’ultimo byte.

Gestione della modalità P2P dei device

Anche per la gestione dei messaggi ricevuti in P2P dai device viene utilizzato lo stesso codice che il server usa per gestire le richieste, in questo modo più messaggi possono essere ricevuti contemporaneamente.

Gestione della connessione al server da parte del device

Ho scelto di controllare se il server ha mandato notifiche al device nei momenti in cui quest’ultimo non deve gestire altre connessioni. I messaggi dal server però vengono ricevuti in ogni caso in modo **sincrono** poiché mi aspetto che ci sia un throughput accettabile e che il server non faccia attacchi DoS.

In caso di disconnessione dal server, il device tenta periodicamente di riconnettersi ma non interrompe le altre operazioni