



FACULTAD DE INGENIERIA

Universidad de Buenos Aires

Taller de Programación Proyecto Cliente de BitTorrent

Grupo Crab Rave - 1er Cuatrimestre 2022

Semorile, Gabriel	105681	gsemorile@fi.uba.ar
Rizzo Ehrenbock, Gonzalo	106475	grizzoe@fi.uba.ar
Nicolini, Franco	105632	frnicolini@fi.uba.ar
Torres Dalmas, Nicolás	98439	ntorresdalma@fi.uba.ar

Índice

Introducción	2
Protocolo BitTorrent	2
Herramientas	2
Arquitectura	3
Comportamiento general y sus componentes principales	3
Comunicación entre nodos de la red	4
Threading	6
Descargas	8
Cargas	8
Interfaz gráfica	10
Logger	12
Conclusiones	13
Bibliografía y Referencias	14

Introducción

Protocolo BitTorrent

Protocolo utilizado para el intercambio de archivos de manera descentralizada usando la arquitectura Peer to Peer (P2P). En una arquitectura P2P, los clientes pueden conectarse entre sí, sin la necesidad de pasar por un servidor centralizado, de esta manera, todos los clientes colaboran entre sí transfiriendo pequeñas partes del archivo en cuestión.

En particular, este sistema de clientes conectados pueden interpretar los archivos con la extensión .torrent. Este archivo contiene datos como la longitud del archivo, cantidad de piezas, e información para conectarnos con el Tracker. El Tracker funciona de orquestador de clientes para el archivo que se quiere descargar, éste nos brindará, a través del protocolo HTTP, una lista de hasta 50 clientes con su dirección ip, puerto e identificador.

Esta red de nodos nos garantizará la transferencia del archivo. Estos nodos se pueden identificar como Seeders (nodo que tiene el archivo del .torrent completo y comparte sus piezas), Peers (nodo que está descargando un archivo y tiene piezas que puede compartir a la red) y Leechers (nodo que está descargando el archivo pero no comparte sus pieza). Entonces una vez me conecte con un nodo y me comparta una pieza, ya puedo compartir esa pieza a quien me lo pida, es decir, me agrego a la red de nodos del archivo en cuestión.

La información del .torrent y la respuesta del tracker están codificadas con el formato Bencoding.

Herramientas

Para el desarrollo de la implementación de un Cliente de BitTorrent Rústico, se utilizaron las siguientes herramientas:

- *Rust 1.61.0*
- *Cargo 1.61.0*
- *Dependencias:*
 - *native-tls 0.2*
 - *chrono 0.4.19*
 - *sha1 0.10.1*
 - *gtk3*
 - *glib*

Arquitectura

Comportamiento general y sus componentes principales

Para la implementación de este Cliente de *'BitTorrent'* comenzamos por leer el *'torrent'* y decodificarlo con nuestras estructuras *'Bdecoder'* y *'Bencoder'*. Luego armamos el *'request'* para el *'Tracker'* con los datos previamente decodificados. Una vez obtenemos la lista de nodos que nos pueden compartir las piezas del archivo, vamos creando nuestros *'Peers'* para comenzar la conexión usando nuestro *'PeerConnection'* y posteriormente pedirle las piezas que necesitamos a través del *'DownloadManager'*. Al terminar la descarga de cada pieza se verificará que sea correcta y se guardará en el directorio establecido. Por otro lado, también vamos a estar escuchando mensajes con nuestro *'Listener'* para poder compartir nuestras piezas con otro nodo de la red. La entidad encargada de esto es el *'UploadManager'*.

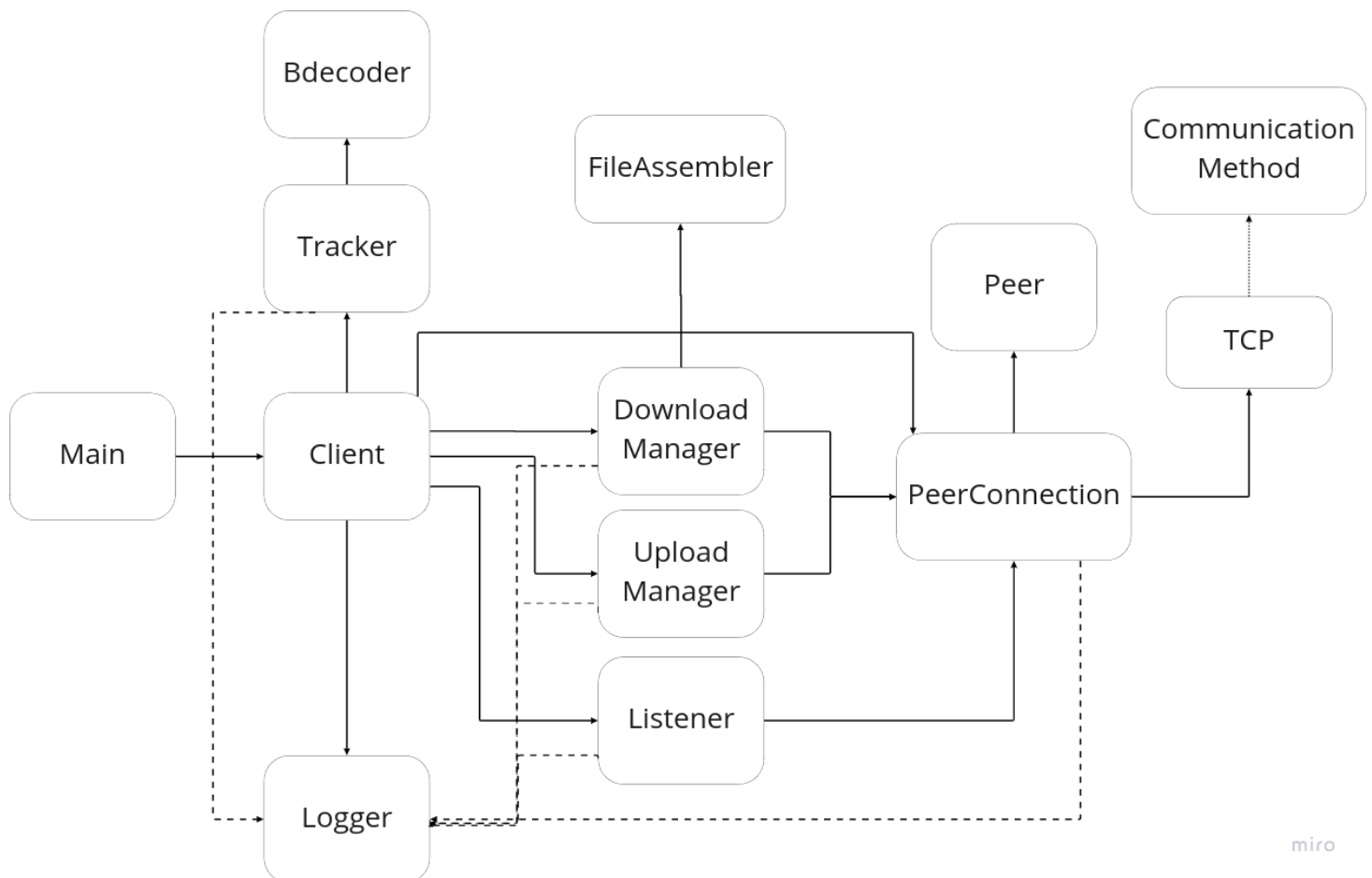


Imagen 1 - Diagrama General

Comunicación entre nodos de la red

En este protocolo para hablar con nodos de la red se necesita un *'id'* y un *'infohash'*. Con estos datos se arma un *'handshake'* con el nodo a interactuar. Una vez terminado satisfactoriamente el intercambio del handshake con el nodo ya podemos comenzar a interactuar con los mensajes P2P del protocolo. También vamos a recibir el *'bitfield'* indicando que piezas nos puede compartir este nodo.

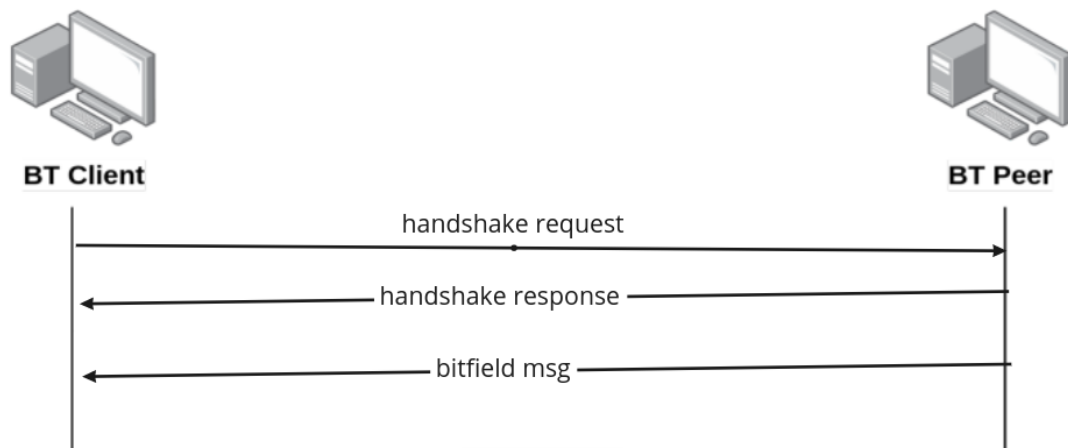


Imagen 2 - inicio de comunicación

Para descargar un archivo, vamos a desbloquear el nodo con un *'unchoke'* y vamos a mostrar interés en al menos una de sus piezas con un *'interested'*, luego aguardamos a que nos desbloquee con un *'unchoke'* y a partir de ahí comienza el intercambio de mensajes *'request'* y *'piece'* para las piezas que nos pueda compartir, respectivamente.

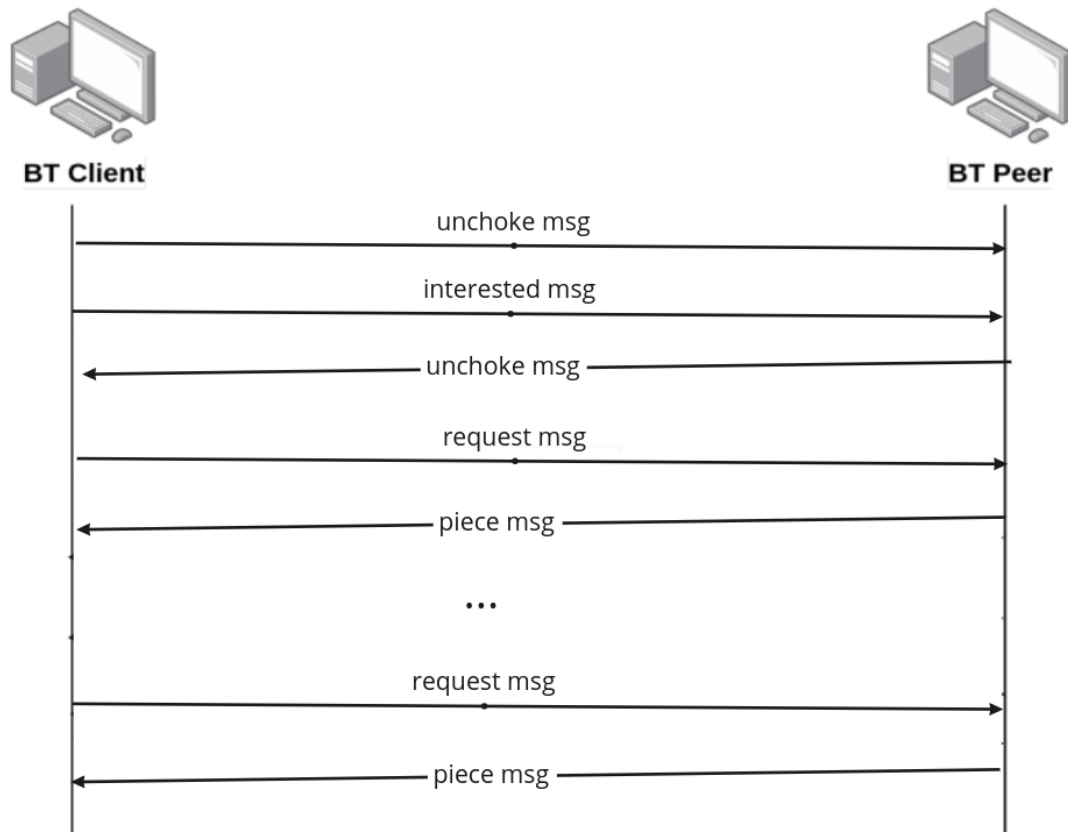


Imagen 3 - descarga de piezas

Threading

Para la comunicación entre las partes, contamos con un thread principal. En el caso de descarga concurrente de muchos archivos se tendrán múltiples clientes, cada uno en un hilo.

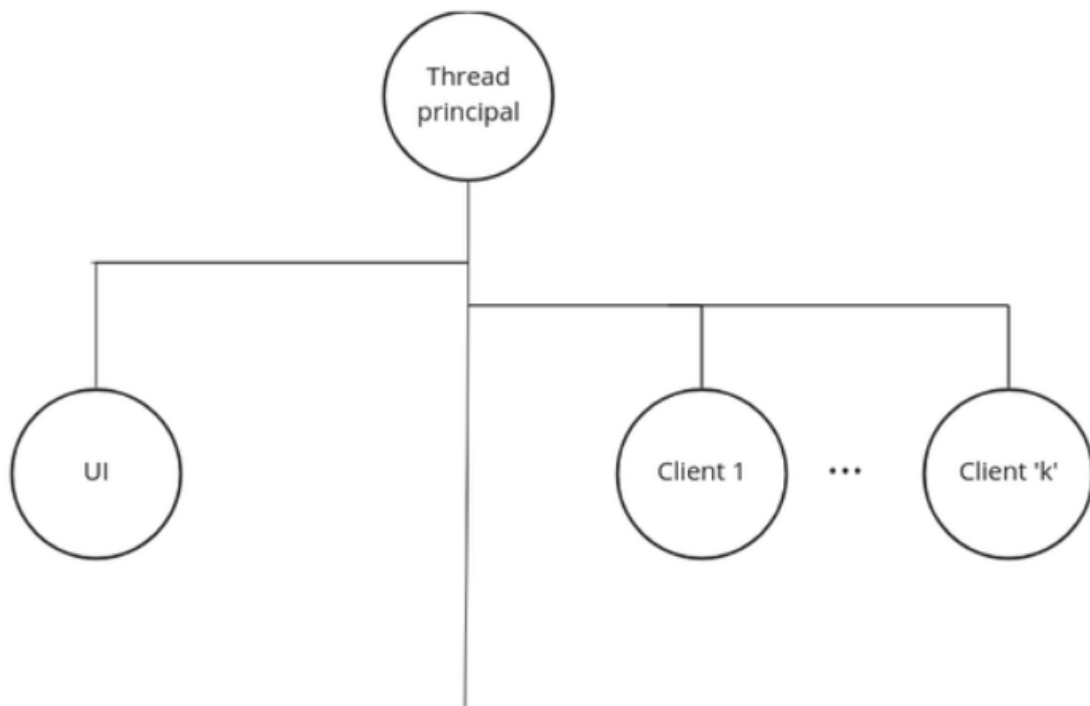


Imagen 4 - Thread Principal

A su vez, cada cliente se comunica con su propio Listener, UploadManager, Logger y DownloadManager. En este último para descargar las distintas piezas se generará un hilo por cada peer al que se solicita y, luego de esperar a que se descarguen todas las piezas, se le hace join a cada thread generado.

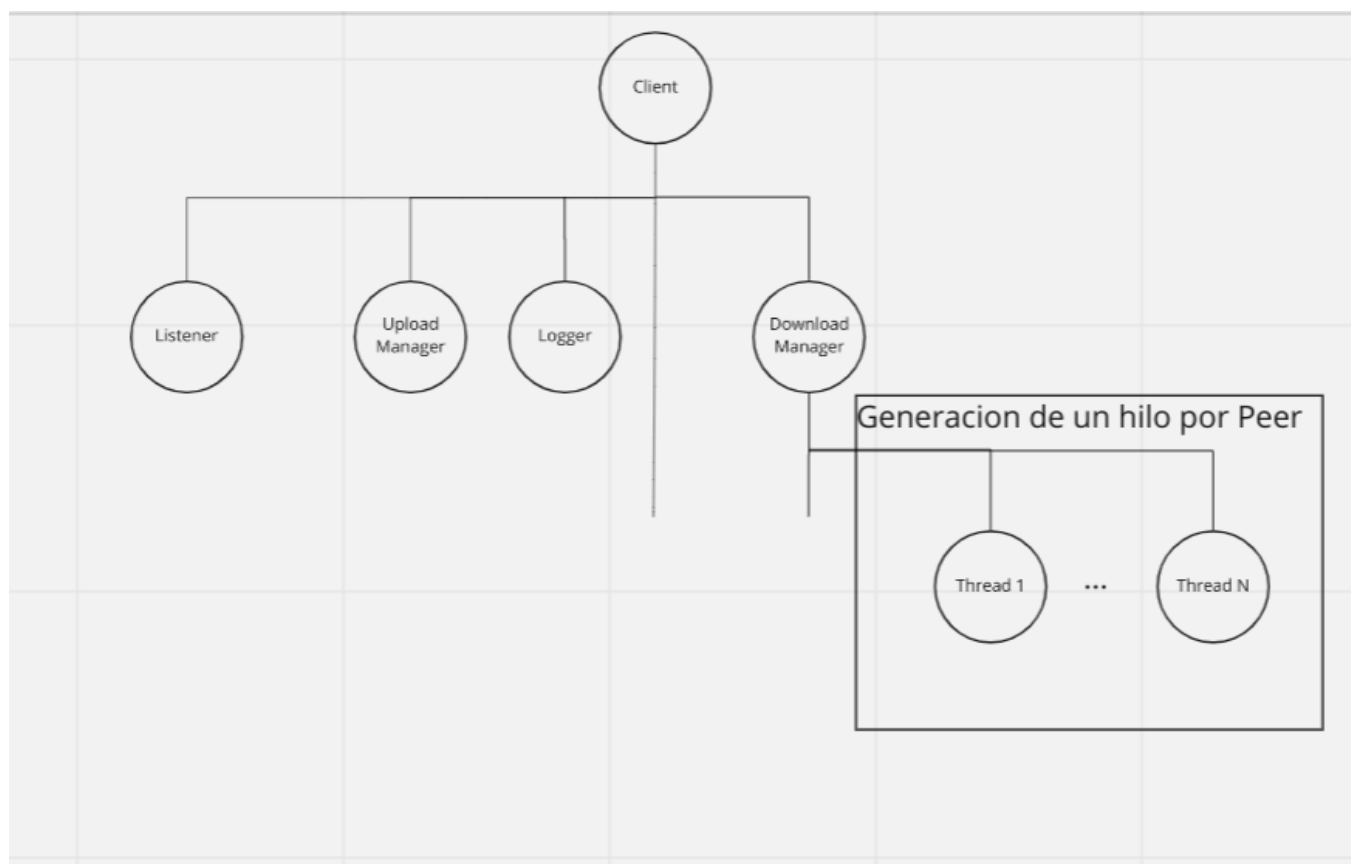


Imagen 5 - DownloadManager y Thread Client

Descargas

Primero para las descargas vamos a realizar las conexiones con *'PeerConnection'* a la lista de *'Peer'* del *'Client'*'. Generamos un hilo por cada peer disponible para descargar cada pieza. Una vez tengo estas piezas verificadas, nuestro *'Download Manager'* llamará al "File Assembler" para que se encargue de ensamblar todas las piezas, construyendo el archivo en el path especificado en el archivo de configuración.

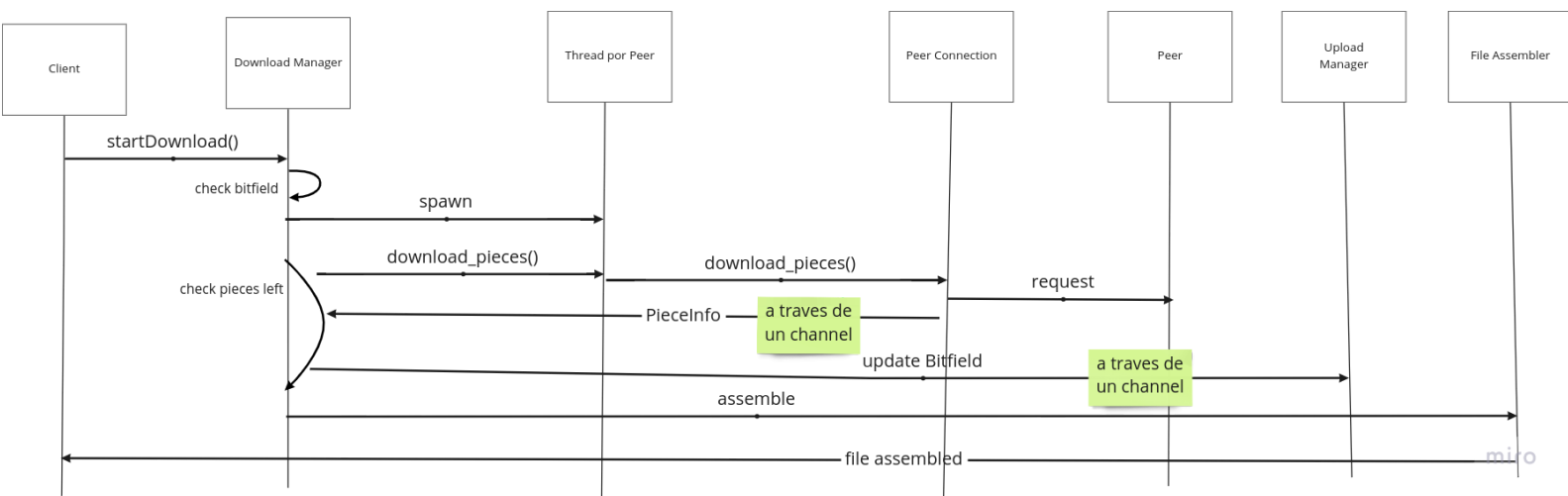


Imagen 6 - descarga de piezas

Cargas

Por otro lado, para las cargas vamos a chequear nuestro *'bitfield'* para saber que piezas tenemos descargadas. En este caso, vamos a tener *'IncomingPeer'* a diferencia del *'Peer'* del *'DownloadManager'*, ya que de antemano no sabemos el *'id'* del nodo que nos quiere hablar.

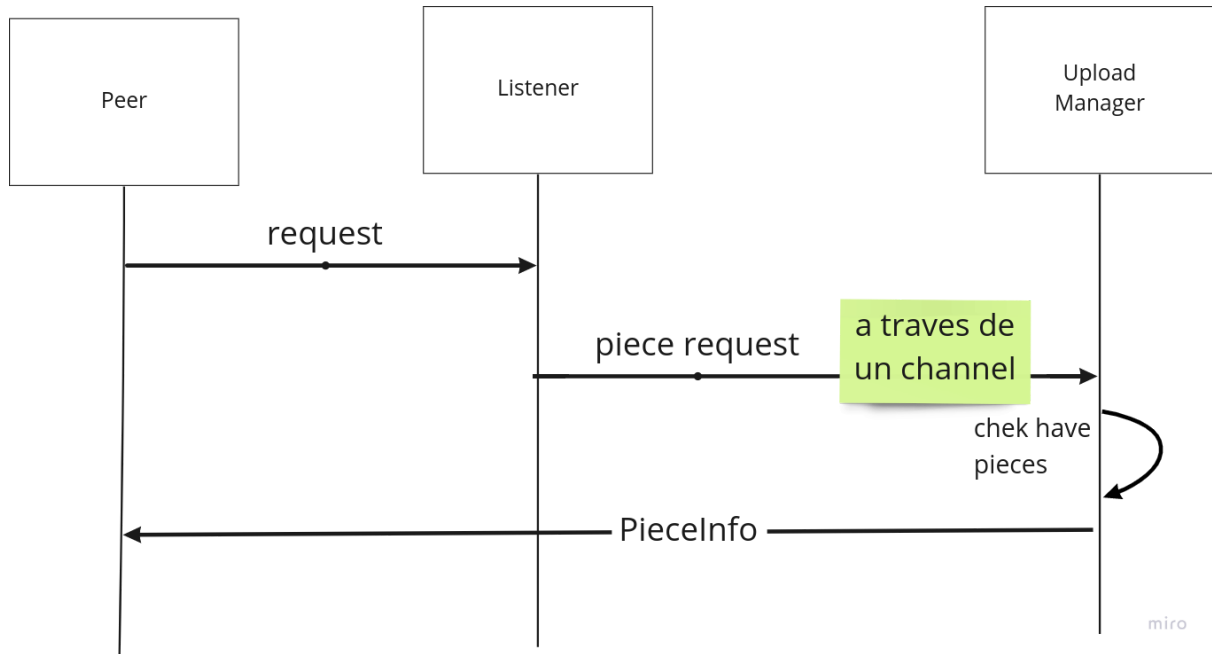


Imagen 7 - carga de piezas

Interfaz gráfica

Para el desarrollo de la interfaz gráfica se usó la herramienta GTK3, y a través de su complemento 'glade' pudimos generar el diseño fácilmente. Al correr el programa lo primero en correr es la UI la cual es la encargada de iniciar al Client en un thread diferente.

Bittorrent Client - CrabRave

ubuntu-22.04-desktop-amd64.iso.torrent

ubuntu-21.10-desktop-amd64.iso.torrent

General Info

Download Details

File Name:

ubuntu-22.04-desktop-amd64.iso

0%

Downloaded Pieces:

49

Total Size:

3 GB

Verified Pieces:

49

Peers Quantity:

50

Active Connections:

8

Imagen 8 - pestaña de información general

En la imagen anterior se muestra la **Pestaña de información general**: la cual cuenta con la información general de los torrents en curso, incluyendo: nombre del archivo, tamaño total, cantidad de piezas, cantidad de peers, además del estado general de la descarga (porcentaje de completitud, cantidad de piezas descargadas y verificadas) y la cantidad de conexiones activas correspondiente a cada torrent.

Bittorrent Client - CrabRave			
ubuntu-22.04-desktop-amd64.iso.torrent		ubuntu-21.10-desktop-amd64.iso.torrent	
General Info		Download Details	
ID	IP	Port	Downloaded Speed
-TR3000-z3jw46001mg6	2a0a:a540:3418:0:d250:99ff:fe98:1ea6	51413	2048 bytes / sec
-TR3000-1e80dp5nkgkz	2a01:4f9:4a:35da::2	51413	1260 bytes / sec
-TR3000-arhnrlas2io9	2602:fdb8:131:4001::c:1	54724	1820 bytes / sec
-TR2940-iz1a3s02i0ib	2a00:1370:8196:2b6:1a31:bfff:fe0d:aeec	60562	1024 bytes / sec
-DE13F0-YRDzC-ATPHc6	2001:41d0:a:2448::1	58276	2730 bytes / sec
-TR2940-sbcsuc03wyo2	2a02:2698:902a:18e:208:9bff:fece:443d	51413	862 bytes / sec
-TR2940-p8uvaml2qw36	2405:9800:bc10:31fd:6600:6aff:fe4e:9e91	51413	1024 bytes / sec
-TR3000-qs253l0egero	2600:6c40:5600:221:d0ad:b6ce:a14c:48ec	51413	1820 bytes / sec
-TR3000-u92ag7nxqcfx	2001:67c:2564:331:a60:6eff:fee8:80b8	51413	1820 bytes / sec
-TR2920-txxuqhqcycujz	2a03:75c0:1e:1a44::1982	51413	2048 bytes / sec
-TR2940-urfua24adcgb	2001:bc8:6010:212:ec4:7aff:fe71:7f0c	51413	2048 bytes / sec
-TR3000-su27aj00v3gy	2003:c9:2f46:f701:3c4a:92ff:fe03:aa47	51413	1024 bytes / sec
-TR2940-yez5l3i62qf1	2a02:1210:4831:9700:ba27:ebff:fe91:60cd	51316	2048 bytes / sec
-TR2940-3hzgvrufsf3	2600:1700:42c6:31f:dea6:32ff:febcb:93c0	51414	2048 bytes / sec
-TR2920-tembyn6h26iy	2a03:75c0:1e:1a44::1982	51413	0 bytes / sec
-TR3000-z3jw46001mg6	2a0a:a540:3418:0:d250:99ff:fe98:1ea6	51413	0 bytes / sec
-TR3000-re0yzuom0e97	2a01:e0a:b2e:3050:a6b1:c1ff:fe0c:98b5	51413	0 bytes / sec
-DE13F0-YRDzC-ATPHc6	2001:41d0:a:2448::1	58276	0 bytes / sec

Imagen 9 - estadística de descargas

En las imágenes de la ventana ‘Download Details’ se muestran las **estadísticas de descargas**. La cual cuenta con todas las estadísticas de las descargas en curso, incluyendo: lista de Peers con ID, IP y puerto de cada una de las conexiones activas, velocidad de bajada y subida de cada conexión, estado del Peer (choked / unchoked, interested/not interested) y estado del cliente en el Peer (choked/unchoked, interested/not interested).

Aclaración: La columna Upload Speed se muestra en 0 debido a que en el momento de la captura de la Imagen ningún Peers nos estaba pidiendo piezas.

Download Details					
t	Downloaded Speed	Uploaded Speed	Peer Status	Client Status	File Name
6	4096 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1820 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1024 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
1	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	0 bytes / sec	0 bytes / sec	Choked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
4	1820 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
0	1820 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1024 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
2	1024 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
0	0 bytes / sec	0 bytes / sec	Unchoked	Not Interested	ubuntu-21.10-desktop-amd64.iso.torrent
1	1024 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
0	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
3	963 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
3	1638 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1489 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1024 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
0	0 bytes / sec	0 bytes / sec	Unchoked	Not Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
1	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
1	2048 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-21.10-desktop-amd64.iso.torrent
4	1170 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent
3	1260 bytes / sec	0 bytes / sec	Unchoked	Interested	ubuntu-22.04-desktop-amd64.iso.torrent

Imagen 10 - detalle de descargas

Logger

Otra entidad importante es el Logger, que nos ayudará a registrar las actualizaciones que vayan ocurriendo durante la ejecución del cliente.

El cliente crea al logger y este a su vez crea un archivo .txt donde se guardarán los mensajes a loggear.

La entidad Logger comenzará a escuchar los mensajes que se reciban

En este ejemplo, cuando se logra establecer una conexión con un peer, *PeerConnection* envía el mensaje "*Peer Connection Established*" mediante el '*channel*' que conecta a este y el *Logger*.

El *Logger* lo recibe y luego lo escribe en el archivo.

Si hay muchos archivos, la descarga concurrente de estos crea un logger por cada uno.

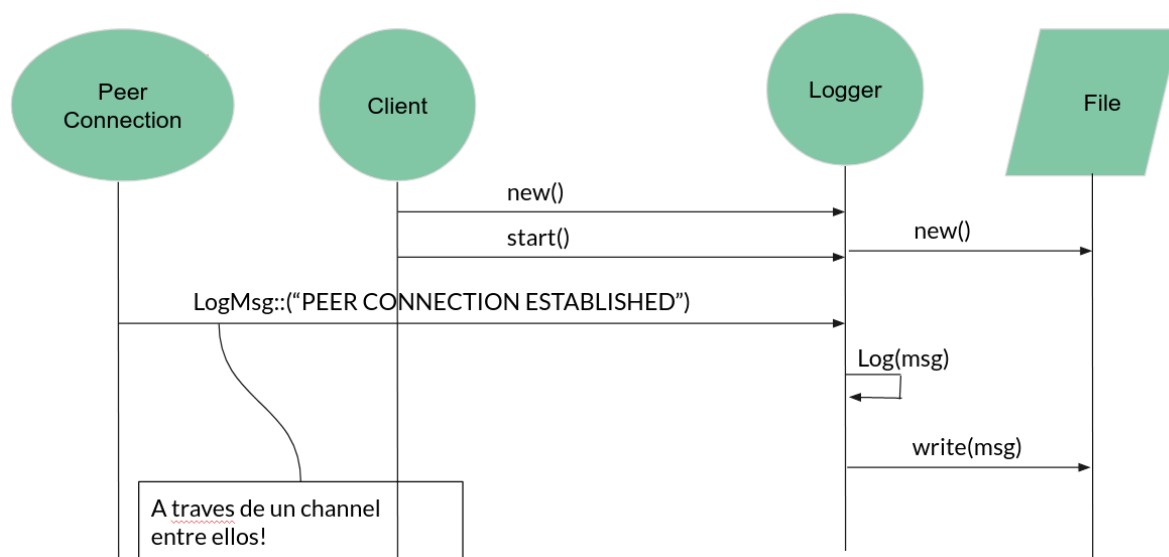


Imagen 11 - Logger

Conclusiones

El desarrollo de este proyecto nos supuso grandes desafíos y un mayor aprendizaje al resolverlos.

Al intentar conectar con el '*Tracker*', descubrimos cómo realizar conexiones y peticiones '*HTTP*'. Con la respuesta de los '*peers*' también encontramos diferencias entre los clientes con ipv4 vs ipv6.

Si bien hasta la entrega intermedia no habíamos implementado el multithreading, una vez implementado el mismo es evidente la diferencia entre las velocidades de descargas entre una sin *'multithreading'* y otra con.

Antes de la entrega intermedia, descargábamos las piezas del archivo una por una, luego la estrategia multithreading nos permitió pedirle varias piezas a varios *'peers'* en simultáneo, logrando así una descarga más rápida del archivo en su totalidad.

Para realizar la interfaz de usuario notamos que lo más conveniente era correr la interfaz y el cliente por hilos separados, conectados estos dos por un *'channel'*. De esta manera, al actualizar alguna información del cliente, se le envía por el *'channel'* a la UI un código que identifica el objeto a actualizar y el valor con el que actualizar. La interfaz lo reconoce y se modifica en consecuencia.

Para lograr una comunicación eficaz entre el cliente y la UI hemos utilizado los *'channels'* especiales de *'GTK'*.

Bibliografía y Referencias

- Especificaciones de parámetros del .torrent: <https://wiki.theory.org/BitTorrentSpecification>
- BitTorrent Developers: <https://www.bittorrent.org/>
- Ubuntu .torrents: https://torrent.ubuntu.com/tracker_index
- Rust Book: <https://doc.rust-lang.org/book/>
- Rust Crates: <https://crates.io/crates/>

- Gtk: <https://github.com/gtk-rs/gtk3-rs.git>
- Glib: <https://github.com/gtk-rs/gtk-rs-core.git>