A Robot Economy for Music Without Any Middleman

Tim Wissel

TU Delft

A Robot Economy for Music Without Any Middleman

by

Tim Wissel

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on XXX at XXX.

Student number: 4460251
Project duration: March 6, 2020 – XXX
Thesis committee: Dr. ir. J. Pouwelse, TU Delft, supervisor

An electronic version of this thesis is available at `http://repository.tudelft.nl/`.

TU Delft

Preface

Preface…

Tim Wissel
Delft, March 2020

Contents

1

Introduction

Most audio streaming services are run by companies, incentivized to make money. They take large cuts of the subscription money from its users. As a result, the artists receive a low compensation. The distributors Spotify, iTunes and Google Play take on average a 25% cut for signed records and 40% cut for unsigned records. This thesis investigates the feasibility and usability of an audio streaming service without a central distributor.

This thesis proposes a solution in the form of a decentralized system which uses a decentralized autonomous organization (DAO) to operate. Listeners, artists and robots form this DAO. The DAO has a shared responsibility for distributing content. In this system, its users (artists and listeners) share audio files and metadata without any middleman. Additionally, users can give donations to artists while the system does not take a cut of these donations. The user can use this system to discover, search and play audio files, targeted at music and podcasts.

Section X describes the design of the system, section Y its implementation and Z its testing results.

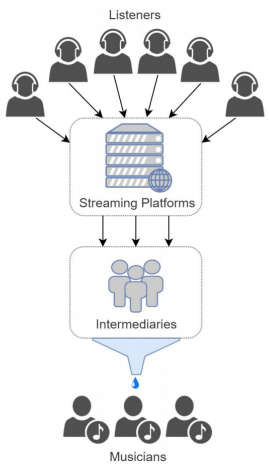

Figure 1.1: Artist compensation inconsistency

2

Problem description

The most widely used music streaming services, with the largest music catalogs, run centralized, proprietary and closed-source software. The companies owning these services have a large amount of power in the music streaming industry because of their scale. The top 5 streaming services have a combined market share of 81%, so this can be regarded as an oligarchy. Because of their power, they can ask high commission fees or lock artists to one platform. As a result, artists receive low compensation. Furthermore, the recommendation and playlist generation algorithms are also a black box for the user. This gives streaming companies curatorial power. At the time of writing, there exists no alternative decentralized and transparent music streaming system with peer-to-peer payments directly to artists. *How can we design and implement a music streaming service that distributes the power from one authority to its users?*

2.0.1. Platformization and gatekeeping

The infrastructure of current internet applications are increasingly moving towards 'platformization'. In essence, platforms are taking control of "the surface on which the market exchange take place" (Andersson Schwarz, 2016) with digital distribution and network effects enabling an increasing centralization of power. This phenomenon is related to IT gatekeeping: tying access of content to a specific internet service. An example of this is the release of the album *The Life of Pablo* in 2016, which was contracted to only be played on one platform, Tidal. In addition, the big music platforms are able to create a digital enclosure of content. This facilitates rent-based monetization of user data, described by Harvey (2002) as 'monopoly rent'. Namely, advertisers can lease a virtual space, to match their advertisements to user profiles. The latest movement in platform accumulation is the monopolization of data. Large scale of data about user interactions with the platform forms a 'monopoly of knowledge' (Innis, 2007). The power of platform companies are raising because platforms, in general, tend towards monopoly (Srnicek, 2017).

In relation to gatekeeping, platforms are now given the task to perform moral judgements on content, for example whether to censor a certain artist. This is controversial as these judgements are no longer in the hands of users but rather in the hands of companies. Recent examples exist such as the disappearance of Li Zhi[1], who published songs about democracy and social issues in China. All of China's main streaming sites removed his songs. In 2019, Apple Music also removed content from their platform by singer Jacky Cheung, who referenced the tragedies of Tiananmen Square in his songs[2].

2.0.2. Intermediaries take a large share

Artists publishing their content on a music streaming service such as Google Music, Spotify and Apple Music receive low compensation, because the intermediaries take a large share. According to Midia Research, the top 5 streaming services control 81% market share (Mulligan, 2020), which can be regarded as an oligarchy. Together these streaming services have the power to ask high subscription fees. After multiple consecutive years of growth, 2019 became the first year in which digital streaming is the single biggest source of revenue for the music industry globally (IFPI, 2020). The global music market also gained a 33.5% growth in paid

[1] https://www.independent.co.uk/news/world/asia/tiananmen-square-china-li-zhi-singer-disappears-anniversary-protests-a8940641.html

[2] https://hongkongfp.com/2019/04/09/apple-music-china-removes-jacky-cheung-song-reference-tiananmen-massacre/

streaming subscribers during this year (IFPI, 2020). This trend is shifting power from decentralized music stores to centralized streaming services.

At the same time, streaming services take a large cut in revenue, and artists are having a harder time making money from music. According to investigations by aCooke (2018) and ReCode (2015), the revenue cut of Apple Music and Spotify is between 25% and 30%. An additional problem is opacity: streaming deals on these platforms remain behind closed doors. For these reasons, massive audiences are needed to generate sustaining profits. An investigation by Bloomberg[3] shows that a whopping 152,094 Spotify subscriber streams generate $100 on average for artists. Consequently, only 0.733% of all acts generate enough revenue for an artist to make a living (Ingham, 2018). The International Federation of the Phonographic Industry states that as of 2018 there exists a "value gap" in digital music streaming, meaning a “mismatch between the value that some digital platforms [...] extract from music and the revenue returned to the music community–those who are creating and investing in music” (IFPI, 2018).

2.0.3. Monopsony power

Monopsony power means that a dominant buyer has the power to push prices down with suppliers. In the context of music, this means that artists have little choice over which platform to publish their music on, because of the dominance of one platform. A few major players in the music industry together form an oligopolistic market. Monopsony power in this area can lead to squeezing the producer side. An example of monopsony power is an event that happened in 2014, between Amazon and Hachette. Amazon, having a large market share on e-books, used its commercial muscle to demand a larger cut of the price of Hachette books it sells. This included for all Hachette books “preventing customers from being able to pre-order titles, reducing the discounts it offered on books and delaying shipment” (Ellis-Petersen, 2014).

Along the same lines, the music streaming oligarchs can use their commercial muscle to demand low pays to artists. Spotify founder and CEO Daniel Ek declared to its investors that the increase in interactions with its in-house curated playlists “puts Spotify in control of the demand curve” [4].

2.0.4. Recommendations and curatorial power

The Big Tech music companies recommend content that best fits their business model, which may be contrary to what is most useful to their customer. The companies can promote or dis-promote content by their choosing. This shows “curatorial power”: the ability to advance own interests by organizing and prioritizing content (Prey, 2020).

Musicians and record labels are increasingly more dependent on landing on Spotify-curated playlists. For example, a study done by the European Commission shows that, for a track to land on the Spotify-curated playlist “Today’s Top Hits”, it will see an increase of $163,000 in revenue (Aguiar & Waldfogel, 2018). 99 of the top 100 playlists are curated by the streaming company. So its recommendation algorithms and playlist curation systems are highly influential.

As the top music streaming services run closed-source software, the inner workings of the recommendation algorithms are opaque to the user. These algorithms, fed by user interaction data, are in some extend also a black box to the company, as they are built using machine learning technologies. However, as noted by (Gillespie, 2014), the impression that algorithms are objective is a “carefully crafted fiction”. Namely, they are altered based on company strategies. Companies are not obliged to explain their algorithms. In the context of recommendations, this leads to a “threat of invisibility": the problem of content regularly disappearing (Bucher, 2018), a phenomenon which is out of the hands of the artist, because of an asymmetry in knowledge over the algorithm workings. Frustrating for artists and labels is also the opaqueness of getting playlisted: it is unclear why “[...] a particular track was placed, or replaced, on a playlist” (Prey, 2020).

[3] https://www.bloomberg.com/opinion/articles/2017-09-25/the-music-business-is-more-unfair-than-ever

[4] https://investors.spotify.com/financials/press-release-details/2019/Spotify-Technology-SA-Announces-Financial-Results-for-Fourth-Quarter-2018/default.aspx

3

Related work

3.0.1. Decentralized audio streaming services

Multiple decentralized audio streaming services exist. Examples are Audius[1], Resonate**?** and eMusic[2]. All of these systems have in common that they save metadata and identifiers of audio files on a blockchain, and save the audio files in an off-chain database. All these off-chain databases are structured like IPFS[3] with a company-run centralized interface between the user interface and the database. For a system to be fully decentralized, this layer should be removed. These solutions are closed source. Moreover, they use their own cryptocurrency to pay their artists which is an unstable income.

3.0.2. Decentralized content delivery networks

Decentralized content delivery networks are being investigated by multiple systems such as VideoCoin[4] and DCDN[5]. Most of these start-ups use blockchain technology and their own-released cryptocurrency as a token to pay nodes that serve the content. This means that the incentive for running a node depends on the value of those cryptocurrencies, so this is an unstable situation for workers.

A fully decentralized audio streaming service requires sharing and streaming audio files over a network of nodes in which any participant can start and run a node. An example of such network is BitTorrent. The challenge with BitTorrent acting as a streaming service is that the requirement from the user perspective is to have low latency for streaming and buffering media files. For each file, the peer discovery algorithm is run, which is a slow-start algorithm. It also relies on having enough seeders per file available.

Torrent files contain a list of chunks, which represent the different parts of the related file. These chunks are called torrent pieces. Flawless streaming of media files over BitTorrent requires a smart algorithm to predict what file is requested next, and what torrent pieces should be loaded. BitTorrent relies on trackers to perform peer discovery. However, trackers are a central point of failure. To make the system more decentralized, a solution using independent trackers and a gossip protocol**?** can be used.

3.0.3. Incentives for file spreading

In a DAO, the party responsible for hosting and spreading of files is not well-defined. To tackle the tragedy of the commons, entities should be incentivised just enough for the system to be sustainable and usable, but no more. An example incentive system is bandwidth tokens**?**.

[1] `https://audius.co`
[2] `https://eMusic.com`
[3] `https://ipfs.io/`
[4] `www.videocoin.io`
[5] `https://www.dcdn.com/`

4

Design

5

Implementation

5.1.

5.2. Metadata storage and discovery

5.2.1. Release blocks

Releases are objects that represent an audio release produced by one or more artists. For example this is an album, an EP, a single, or a podcast. Release objects are stored on the TrustChain, and are exchanged between peers that are part of the MusicCommunity (see **??**). These objects are structured as shown in 5.1. The *magnet* property contains a magnet link which holds all additional information about the contents of the Release, such as the torrent file list, size and tracker URLs. The *publisher* property contains the public key of a Bitcoin wallet which is owned by the creator of the Release. This public key is an identity of the owner of the record (either the artist, band or label) and can be seen as a proof of the identity of an artist or collaboration between them.

5.2.2. Searching

The MusicCommunity extends the TrustChainCommunity with additional methods *performRemoteKeyword-Search* and *onKeywordSearch* which allows peers to search for content using keywords, both locally and remotely. These methods test each *title*, *artists* and *date* field of each Release block against a `contain(keyword)` filter. This means: if the keyword is contained in one of those values, it is added to the results.

When the user performs a search, the local database is filtered first to find matches. If there are less than x results, the device sends a KeywordSearchMessage (see X) to a maximum of P known peers in the MusicCommunity. This asks neighbours to inspect their local database to find matches for the same query. Upon receival, the peer subtracts 1 from the time to live (TTL) field. If the peer finds a match, it sends the corresponding TrustChain block directly back to the original asker, appointed by the *origin* field. This contains the IPv8 *peer ID* (Skala, 2020) of the peer that initiated the search. Otherwise, if the time to live (TTL) is more than 1, the peer forwards this KeywordSearchMessage (see **??**) to other peers. Our keyword search is implemented with default values of $T = 1, P = 20, x = 5$ (where T is the TTL).

5.3. Networking

5.3.1. Torrent creation and sharing

We use the JLibtorrent implementation by Frostwire[1] of BitTorrent to interact with BitTorrent peers. In our application, a user can select local audio tracks, after which a torrent file is generated with the corresponding metadata. Alternatively, the user can paste a preexisting torrent magnet link to use for the Release block, as shown in 5.4. The creation of torrents by the app happens as follows. First, the user presses the *Select Local Audio* button in the release creation dialog (see 5.3). Afterwards, the user selects one or more tracks to add (see 5.4). In the background this creates a torrent file (using the Ttorrent library[2]), which is stored on the

[1] `https://github.com/frostwire/frostwire-jlibtorrent`
[2] `http://mpetazzoni.github.io/ttorrent/`

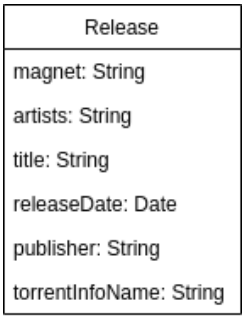

Figure 5.1: Release blocks structure as seen on TrustChain

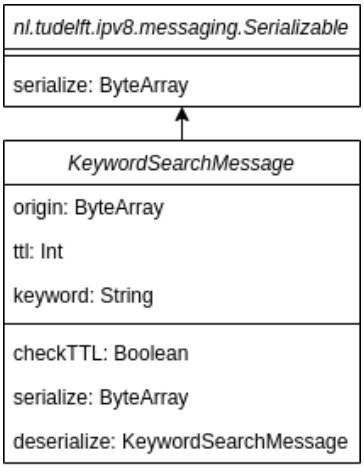

Figure 5.2: KeywordSearchMessage object sent over IPv8 in MusicCommunity

mobile device, and added to the ContentSeeder (see 5.3.2). Finally, by using the computed infohash and file list, the magnet link of the torrent file is created and added to the *magnet* field of the Release block.

By default, the torrents created using MusicDAO are marked as 'trackerless' meaning that torrent peers are found using a distributed hash table (*Distributed Hash Table*, 1981-2019) (specifically, mainline DHT[3])instead of centralized trackers. This is to keep the app independent on connectivity on trackers and pushes towards a fully distributed system.

In addition, the app enables the local peer discovery (LPD) (8472, 2015) functionality of BitTorrent. This allows for finding peers and transmitting torrent pieces over local area network. This results in higher transmission speed and lower latency for content that is stored in the cache of devices nearby. For example, if device A is looking for track X, and device B has this track cached and is active on the same local area network, this track can be found and buffered quickly from A.

5.3.2. Content seeding

Seeding of content is implemented using a simple continuous mechanism. On startup, the ContentSeeder class (see X) spawns a background thread which iteratively scans all torrent files in the local cache directory of the app. There is an upper threshold for how many torrents are selected for seeding. This threshold T is currently set to $T = 10$. The ContentSeeder uses a last-in-first-out heuristic: the top T most recent created/received torrent files are seeded.

5.3.3. Caching

The app implements both caching of tracks and metadata, to enable fast browsing and playback of tracks. As described in 5.3.1, only magnet links are shared on the TrustChain blocks. When the user opens a playlist for the first time, torrent metadata is fetched from peers for the corresponding magnet link. Once this metadata is received, it is written to a new .torrent file in the *cache directory*[4] of the app. This can then be picked up by the ContentSeeder (see 5.3.2). The tracks are also stored the cache directory, in subdirectories identified by the *name* (Cohen, 2008) property of the torrent metadata.

5.4. Music Player and Streaming

Playing music is implemented using ExoPlayer 2.10[5]. This music player library is suitable as it allows for playing tracks that are partially loaded, which allows for streaming.

[3] `https://www.libtorrent.org/dht_extensions.html`
[4] `https://developer.android.com/training/data-storage`
[5] `https://github.com/google/ExoPlayer`

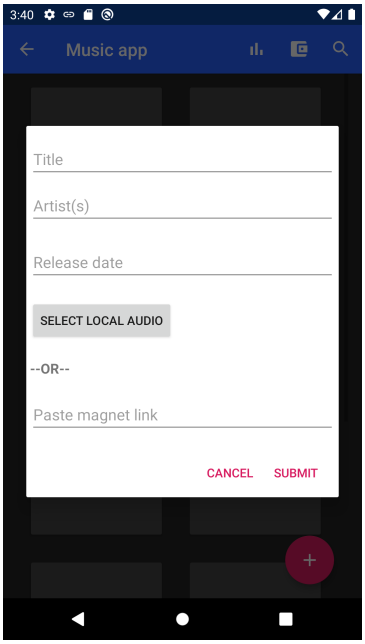

Figure 5.3: Dialog for creating and publishing a new Release

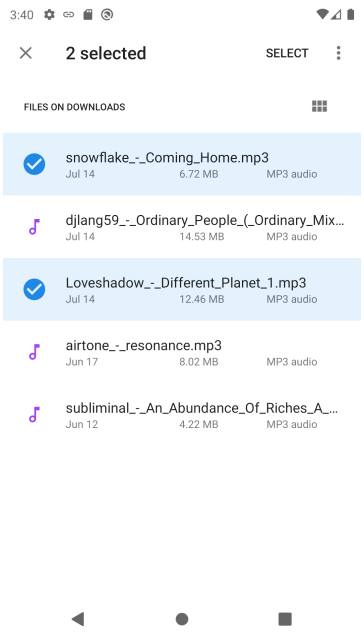

Figure 5.4: Selecting local tracks for creating a new Release

5.4.1. Priority handling

To provide the user with selected content as soon as possible, we implemented a system which sets priorities on certain tracks and torrent chunks. This uses the piece priorities system in libtorrent, which range from the integers 1 (normal) to 7 (highest) (see libtorrent Manual[6]). When a user selects a track, this track is given a high *file_priority* of 5. This is a higher priority than that of other files, so JLibtorrent asks actively for peers who have this file. In addition, the first 5 pieces of the selected track are given a *piece_priority* of 7, so that the first seconds of the track are buffered quickly and the user can start streaming early.

5.4.2. Seeking and buffering

The music player tries to play the track once a satisfactory portion is loaded. This is implemented as follows. Upon seeking a part of the track, the torrent piece index on which the cursor is located is calculated. Afterwards, the piece at this index and the 5 consequent pieces are given a **piece_priority** of 7, which is higher than the other pieces. Once at least 2000 ms from the cursor is buffered, the music player starts playing the track. This way the user can start playing the portion of the track they are most interested in quickly.

5.5. Donations and payments

5.5.1. Bitcoin RegTest network

We created a public Bitcoin RegTest environment[7] to test peer-to-peer Bitcoin donations and payments. This creates a new 'clean slate' Bitcoin blockchain and allows for full control over the chain and miners. This enables a test environment that is useful for experiments, as we can tweak the block generation speed and read all transactions registered on the closed-off blockchain.

5.6. Interface

5.6.1. Playlist overview

The playlist overview screen, as shown in 5.6 is the screen that is first shown upon starting the MusicDAO. Here the user is presented a list of playlists, loaded from their local database TrustChain instance. Currently, each Playlists fragment corresponds to exactly one Release block (see ref). In the background runs an iterative process which checks whether new playlist content is found, and re-renders the view if necessary. The

[6] `https://www.libtorrent.org/manual-ref.html#file-format`
[7] `https://developer.bitcoin.org/examples/testing.html#regtest-mode`

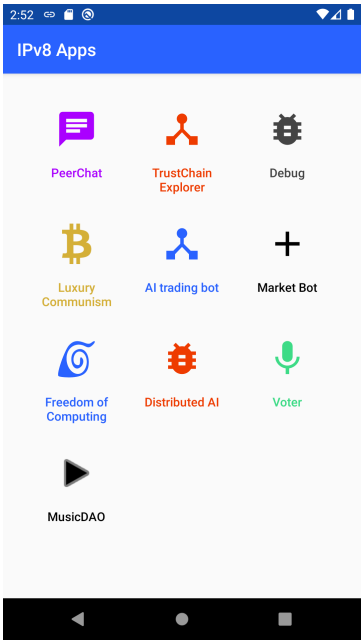

Figure 5.5: The app is integrated as a mini-app in the IPv8 Superapp catalog

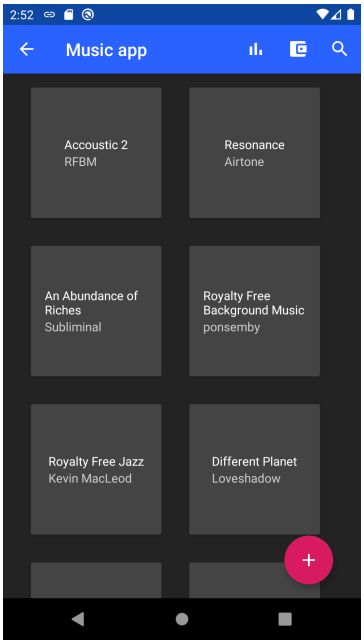

Figure 5.6: The playlist overview screen, which is the entrance screen

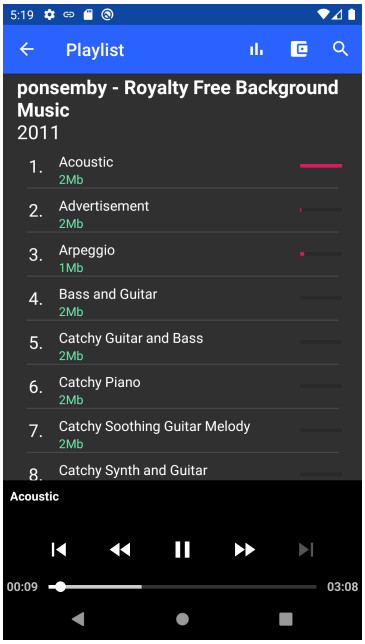

Figure 5.7: Playlist fragment, showing all tracks of one Release

playlists are sorted on their torrent swarm health in ascending order.

5.6.2. Playlist fragment

A Playlist fragment interacts with exactly one Release object. The playlist fragment displays its list of tracks and other metadata, such as the title and artists of the Release (see 5.7). For each track the file size is displayed, and a loading indicator on the right side. This shows, in real time, how much of the track is downloaded. In the example of 5.7, the first track is fully loaded.

5.6.3. Wallet

Each device participating in the MusicCommunity is given a private/public wallet identity upon installation of the MusicDAO. The wallet interface (5.8) shows synchronization status of the RegTest network (see 5.5.1). Once the wallet is fully synchronized with the blockchain, the private key and balance are displayed as shown in 5.9.

Upon browsing a playlist, if the *publisher* field is present in the corresponding Release object, a donation button is displayed as shown in 5.10. When pressing this button, the user can select an amount and make a direct donation to the artist or band, in the form of a bitcoin transaction from their wallet. The user enters a value in USD and the corresponding amount in bitcoin is calculated and shown when this field is edited. This is implemented using the XChange library[8] connecting to the Binance trading platform[9]. After confirmation, the transaction is registered on the RegTest network (see 5.5.1).

[8] `https://github.com/knowm/XChange/releases/tag/xchange-5.0.1`
[9] `https://www.binance.com/en/trade/BTC_USDT`

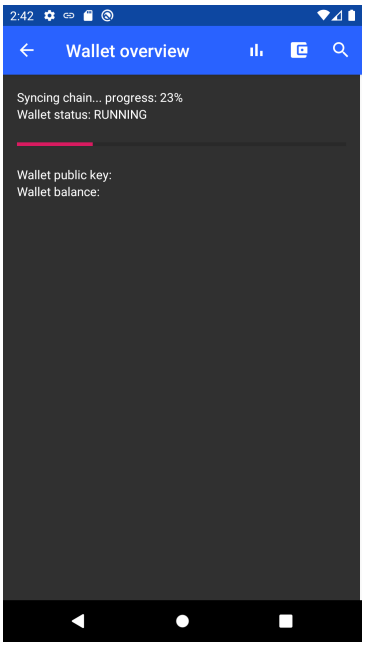

Figure 5.8: Synchronizing with the Bitcoin RegTest environment blockchain

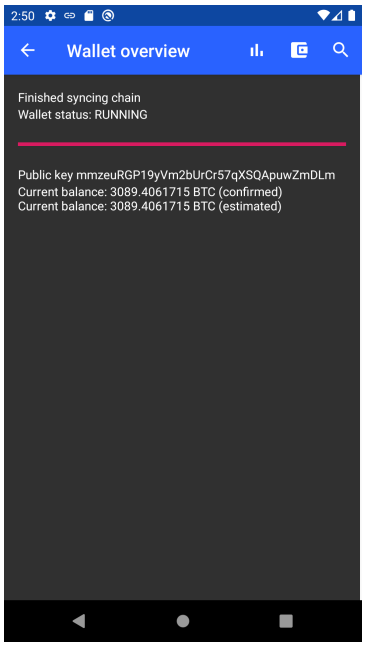

Figure 5.9: Wallet overview and balance after synchronizing

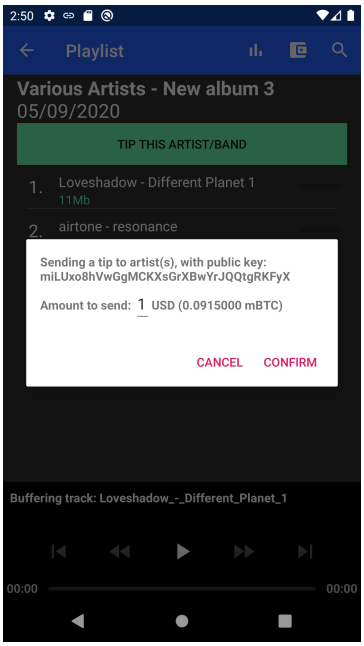

Figure 5.10: Sending a tip to an artist or band

6

Conclusion

Bibliography

8472, T. (2015). *Bittorrent local service discovery.* Retrieved September 14 2020, from http://www.bittorrent.org/beps/bep_0014.html

aCooke, C. (2018). *Dissecting the digital dollar.* London.

Aguiar, L., & Waldfogel, J. (2018). *Platforms, promotion, and product discovery: Evidence from spotify playlists* (Tech. Rep.). National Bureau of Economic Research.

Andersson Schwarz, J. (2016). Mastering one's domain: some key principles of platform capitalism.

Bucher, T. (2018). *If... then: Algorithmic power and politics.* Oxford University Press.

Cohen, B. (2008). *The bittorrent protocol specification.* Retrieved September 19 2020, from https://www.bittorrent.org/beps/bep_0003.html

Distributed hash table. (1981-2019). Retrieved September 14 2020, from https://encyclopedia2.thefreedictionary.com/Distributed+hash+table

Ellis-Petersen, H. (2014). Amazon and hachette end dispute on ebooks. *The Guardian.* Retrieved from https://www.theguardian.com/books/2014/nov/13/amazon-hachette-end-dispute-ebooks

Gillespie, T. (2014). The relevance of algorithms. *Media technologies: Essays on communication, materiality, and society*, *167*(2014), 167.

Harvey, D. (2002). The art of rent: globalisation, monopoly and the commodification of culture. *Socialist register*, *38*.

IFPI. (2018). *Global music report 2018.* International Federation of the Phonographic Industry London.

IFPI. (2020). *Ifpi global music report 2020 - the industry in 2019.* International Federation of the Phonographic Industry London.

Ingham, T. (2018). The odds of an artist becoming a "top tier" earner on spotify today? less than 1%. *Music Business Worldwide*, *25*.

Innis, H. A. (2007). *Empire and communications.* Rowman & Littlefield.

Mulligan, M. (2020). Music subscriber market shares q1 2020. Retrieved from https://www.midiaresearch.com/blog/music-subscriber-market-shares-q1-2020

Prey, R. (2020). Locating power in platformization: Music streaming playlists and curatorial power. *Social Media+ Society*, *6*(2), 2056305120933291.

ReCode. (2015). *Here's what happens to your $10 after you pay for a month of apple music.*

Skala, M. (2020). *Technology stack for decentralized mobile services* (Unpublished master's thesis).

Srnicek, N. (2017). *Platform capitalism.* John Wiley & Sons.