Delft University of Technology

Contextproject - Blockchain

An Embedded, Interactive Explorer for the Tribler Network

■-■-■- BlockchainBoys -■-■-■–

Yinghao Dai Max van Deursen Geert Habben Jansen
Bram van den Heuvel Ruben Keulemans Tim Speelman

supervised by
Martijn Gribnau, Stefan Hugtenburg, Johan Pouwelse and Alberto Bacchelli

June 22, 2017

Abstract

test

Write ab- stract

Contents

1 Introduction **1**
1.1 The Tribler Project . . . 1
1.2 Goal of this project . . . 2
1.3 End-user's requirements . . . 2

2 Product description **3**
2.1 An Overview of the network explorer . . . 3
2.2 The features not implemented . . . 4

3 Code quality description **5**
3.1 Legacy code base . . . 5
3.2 Our code quality . . . 5
3.3 Shipping our code within Tribler . . . 6

4 Components of the Network Explorer **7**
4.1 Front-end code structure . . . 7
4.2 Back-end code structure . . . 8

5 Evaluating Usability and Awareness **9**
5.1 Initial usability . . . 9
5.2 Feedback from the product owner . . . 9
5.3 Evaluation method . . . 9
5.4 Results . . . 10
5.5 Conclusion . . . 10
5.6 Discussion . . . 10

6 Evaluation of the functional modules **12**
6.1 Continuous Integration . . . 12
6.2 User testing didn't break it . . . 12

7 Further development of the product **13**
7.1 Actual Trust™ 13
7.2 More about which nodes are relevant and which aren't 13

A Evaluating Usability and Awareness **14**
A.1 Debriefing questions 14
A.2 User feedback 15
A.2.1 Feedback gathered in the first round 15
A.2.2 Feedback gathered in the second round 15

Chapter 1

Introduction

1.1 The Tribler Project

Tribler (Pouwelse, 2008) is an open-source BitTorrent client for anonymous peer-to-peer file sharing. It is fully decentralized, protecting its users against censorship, lawyer based attacks and government control. Like many decentralized systems, it suffers from a problem known as the Tragedy of the Commons. As described by Hardin (1968), individual users of a shared unregulated resource may act only in their self-interest, posing a threat to the existence of the resource. In the context of file sharing, the resource is kept alive by users uploading their files. Users who do not contribute are called free-riders. In order to protect the resource, systems must find a way to identify them and deny service.

Since Tribler users are anonymous and creating a new identity is virtually free, identifying free-riders is more difficult. Tribler uses MultiChain (Norberhuis, 2015) to capture interactions between users. This establishes the notion of reputation of a peer in the network. Although users cannot directly tamper with these interactions, it is susceptible to Sybil attacks. In this attack a user creates a set of fake identities, and fakes transactions between them, to improve his own reputation. In an attempt to detect these attacks, Otte (2016) has implemented the algorithms Net-Flow and Temporal PageRank into Tribler. Based on the aforementioned technologies, the Tribler software determines a trust level of peers. A peer is considered trustworthy if it has contributed sufficiently. Based on this trust level the software grants or denies service to the peer.

However, experience shows that using a complicated algorithm to determine the trust level, is not always trustworthy itself. Therefore, the Tribler developers decided to use a simple metric in order to decide whether a user should be prevented from downloading content: if the balance (difference) between uploaded and downloaded data gets too negative, i.e. exceeds a given threshold, then the user will be identified as a free-rider. This means that he or she has to start uploading before being able to download again.

1.2 Goal of this project

As most of the Tribler users are unaware of these technologies, they may not understand why their service is being (partially) denied. This may leave them frustrated, and abandon the software instead of improving their contribution to the network.

This gap between the Tribler software and the person using it must be bridged. To this end, we wanted to develop a Network Explorer that shows users exactly what is going on around them, what their reputation is, how that came to be, how they can improve it, but most importantly, why they should improve it.

1.3 End-user's requirements

Since the product is an extension of Tribler, its users are existing Tribler users. Our goal is to offer existing users a new feature that enhances their Tribler experience. Current users need to understand how they obtained their current balance and how this affects their download speed. Moreover, they need to understand how to improve this balance. Users only contribute to the network once it is in their short-term interest (Hardin, 1968).

This understanding is furthered by allowing these users to gain insight into the network of their peers. This is done by showing, for example, the calculated balance of a node in the network, and relations between nodes. The users are then able to compare these different balances.

For this to work, the users need to understand what they are seeing in the Network Explorer. Therefore, there needs to be a help page with a clear description of what the different components represent. This has to be shown in such a way, that an average user without a background in computer science engineering can easily understand. While doing this, we have to ensure that the user experience does not decay. For instance, the start-up time of the Tribler application must not be negatively affected.

Moreover, seeing so much information about the network could make users worried about their own anonymity. We have to make sure that users do not doubt about this matter, and become fearful of using Tribler.

Chapter 2

Product description

When an end-user downloads and/or uploads files using the Tribler network, it exchanges data with other Tribler users. In the background, Tribler keeps track of how much data was exchanged and between whom, not only of the end-user but of all peers it can find. This way it forms an ever-growing picture of the Tribler network and the traffic between users.

2.1 An Overview of the network explorer

The network explorer is simply a view on the gathered data. It displays all users, their upload and download totals, and their connections to other users. It provides the end-user with valuable insight about his or her position in the network, and what is going on.

Users are represented as circles with a short 'anonymous name'. The circles are connected with lines, which represent content exchanged between users. An intuitive representation of the data is given through visual cues such as colors, circle size and line thickness. When the end-user hovers over these circles and lines the exact quantities are displayed in a sub-window called the inspector.

Tribler judges its users based on these data and determines when to stop uploading to a specific user; i.e. when its ‘reputation’ is too bad. Determining a fair metric for reputation is difficult, but Tribler currently uses ‘balance’ for this. The balance is a user’s total upload minus download. Reputation is presented through a color; a user with a poor reputation is colored red, a good user (”toffe peer”) is colored green.

Displaying all known users in a single view would become impossible as the gathered data grows. To this end the network explorer only shows a portion it. It focuses on a single user (initially the end-user), centered in the view, and displays its peers in a ring around it. A second ring contains all peers-of-peers. This is called a ‘radial layout’. When a user has a lot of peers, only its biggest exchanges are displayed.

Any of the users can be clicked, which shifts the focus to that user. The clicked user's circle animates toward the center and its known peers and peers-of-peers are revealed on their respective rings. This way the end-user can explore the network and inspect peers who are more than two steps away. By means of helping the end-user find the way back to his or her own node, a 'Back To You' button is provided which triggers a step-by-step animation navigating all the way back.

2.2 The features not implemented

The original vision of the network explorer was that it displays 'trust'. During the development process it turned out that this notion of trust is not clearly defined by Tribler. It is a very difficult concept to grab and should involve a lot more than the balance of a user. This metric is not resistant against Sybil attacks as mentioned in the Product Vision, for example.

To avoid confusing the end-user, the network explorer should display the same metric on which Tribler bases its judgment: balance. The Tribler developers have plans to elaborate on this, implementing a more advanced measure of trust. Such a metric may depend on the number of peers a user is connected with and what their reputation is; a user who only exchanges data with bad-users might be suspected of performing a Sybil attack.

If Tribler would move towards such a metric, this would justify the use of different layouts. Such layouts could more clearly indicate a notion of 'connectedness' between two users or on how the reputation of a user influences that of its peers.

Chapter 3

Code quality description

3.1 Legacy code base

The Tribler code base has a lot of issues. This issues include bad documentation or no documentation at all, very long methods and classes, inconsistent naming, old Python version usage and untested code. Because the Network Explorer is an addition to Tribler, and uses and extends its REST API, database and database communication classes, we are enforced to work with this legacy code base. This has important implications including investing time in understanding the code we are using and extending, writing maintainable code without proper examples while being consistent with the conventions (if existing). Furthermore we constantly need to rebase our Tribler fork with the Tribler repository. This takes a lot of time because it breaks compatibility sometimes. The above distinguishes the Network Display project form the other context projects, were software is developed from scratch, without the need to integrate it into a legacy code base.

3.2 Our code quality

Working Agile using the SCRUM framework helped us writing high quality code. Each morning in the daily stand up group members indicated issues and the need for help. Pull requests were kept small and we used a "maximal one pull request at a time per person" policy. Per pull request group members reviewed the introduced functionality and code quality. Changes were adopted if needed. This helped us writing high quality code with shared code ownership.

3.3 Shipping our code within Tribler

Chapter 4

Components of the Network Explorer

The Network Explorer consists of modules in both the back-end and the front-end. These modules communicate through a REST API.

4.1 Front-end code structure

The front-end consists of a single page: the Network Explorer page. This page uses a Chromium based browser engine to display the web page containing the actual Network Explorer view. The web-page is built up of a single HTML file which loads all CSS and JavaScript modules responsible for drawing. It loads D3.js, a JavaScript library used for drawing and animation.

When the focus is moved to a user, the front-end makes a request to the back-end. It requests the clicked user as a 'focus node' and asks for its first and second level neighbors. When the response comes back, the DataProcessor does the necessary data conversion required by other front-end modules.

The processed data contains nodes (users) which must first be positioned. A positioning algorithm uses breadth-first traversal to make a tree from the graph. It then divides the space on the rings over all nodes. The animating of nodes to their desired positions is then done using D3 forces. A custom built 'torque force' ensures the nodes move along their circles and not in a straight line.

A set of drawing modules takes care of drawing the users and connections as circles and lines. It binds to mouse events and ensures the correct data is displayed and the correct elements are highlighted.

A separate module is responsible for guiding the user back through the nodes he or she clicked using animated markers and careful timing. This ensures the user does not lose orientation.

4.2 Back-end code structure

The back-end holds the database containing all known transactions. It also holds the Crawler module, responsible for gathering transactions from other users. These modules are part of the Tribler project and are outside of the scope of the Network Explorer.

The Network Explorer uses the database to fill a custom 'aggregate' database, listing all exchanges between users and the total amount of traffic between them.

An API request comes in at the API endpoint. This subsequently calls a module which is responsible for gathering the data from the database. It performs queries on the database, listing all nodes related to the requested focus node and limiting them to a specified number if necessary. It continues to fetch the peers-of-peers for every requested neighbor level. The combined data is then encoded into a JSON response and sent back to the requester.

Chapter 5

Evaluating Usability and Awareness

5.1 Initial usability

5.2 Feedback from the product owner

5.3 Evaluation method

On the basis of the gathered information mentioned above, we set up a user experiment in order to test our product against usability and awareness. Using this test, we wanted to check whether users, with or without a computer science engineering background, could easily understand, use, and know the importance of our product.

When choosing the target audience for participating in our test, we took into account the likelihood that a given person would use Tribler. After all, testing a Tribler component on a person that will never use Tribler, makes little to no sense. Therefore, we asked five people from another groups in our context, three people from other contexts in our course, and two mathematics students from the Leiden University.

Our evaluation was performed in iterations, which meant we executed the test, thereby generating results, evaluated and analyzed these afterwards, before improving our product and starting a new round of evaluation, executing another test, et cetera.

The test execution procedure was as follows. We started explaining the procedure to the participant, thereby stating that we were recording the audio during the tests, and would annihilate all recordings within ten working days. In fact, only one of the audio recordings had been listened to afterwards, as we made useful notes during the tests as well. Moreover, we explained that we wanted the participant to "think aloud" (and how to do this) during the usage of our product, and that we would ask some additional questions afterwards. At the end, if the participant wished to, we could explain more about Tribler and our product, so that the participants would not go home with the unease of having unanswered questions.

After this, the execution of the test took place, exactly as described in the paragraph above. That means we did a minimal-risk test, which we verified using the ethics checklist of the TU Delft Human Research Ethics Committee. We did annihilate the recordings made in the first round, and even refrained from making recordings in the second round, as we felt that the notes sufficed.

This decision followed from the analysis that we made after the first round. When setting up the test procedure, we decided that one person would accompany and question the participant, and a second person would make notes. However, there was a possibility that one of us would miss information or interpret the saying differently, which meant having a recording and being able to listen another time, would be a blessing. However, as this happened only once, this did not outweigh the risks of recording for the participants anymore. Therefore, we decided not to record the audio in the second round.

5.4 Results

Of course, the user test procedure led to results, in the first as well as the second round.

Write something better

5.5 Conclusion

Based on the test results, we can conclude that users can indeed quite easily understand and use our product. This was the main encountered problem during the first round (the comprehensibility for users that are not familiar with Tribler or torrenting), and had been greatly improved before the second round. The awareness of our product's importance, however, turned out to be slightly insufficient in the second round as well. After the improvements of the second round, we believe this issue has been addressed. Unfortunately, there is no third round of user testing in order to verify this.

5.6 Discussion

Having such a third round, would naturally have been an improvement. Moreover, we could have asked other groups of participants than only university students, since they are potential Tribler users as well. Having more participants anyway, would have helped as well, in creating a better view of how the users appreciate our product.

The main limitation of our evaluation procedure, was the procedure itself. By explaining the whole procedure to the participants at the beginning, they feel that everything is done so ‘officially’, which can cause a certain perception of pressure. As we all know, people can behave differently under pressure, either positively or negatively. For instance, we saw that almost every participant clicked on the ‘Help’ button and read all information carefully, whereas we know (not only from personal experience) that users tend to scroll through such manuals really quickly. In order to have a view that reflects reality as accurately as possible, we might have to track users’ behavior when they are comfortably in front of their home computers, without them knowing. It goes without saying, however, that this method would have been highly unethical.

Chapter 6

Evaluation of the functional modules

6.1 Continuous Integration

6.2 User testing didn't break it

Chapter 7

Further development of the product

7.1 Actual Trust$^{\text{TM}}$

7.2 More about which nodes are relevant and which aren't

Appendix A

Evaluating Usability and Awareness

A.1 Debriefing questions

- Usability
  - Did you notice you can click on a circle? What happens when you do so?
  - Did you notice you can hover over circles and lines? What happens when you do so?
  - Did you notice the 'Back To You' button? What happens when you click it?
  - Did you notice the 'Help' button? What happens when you click it?
  - Did you like using this application? Please explain why.
- Awareness
  - Describe what you have just seen.
  - What does a circle represent, what does a circles text, size and color mean?
  - Is it clear you are a 'part' of the network, how is this visualized?
  - How can you influence the color of your own circle?
  - How are the colors of the other circles influenced?
  - What does a line represent, what does a lines size and divider mean?

A.2 User feedback

A.2.1 Feedback gathered in the first round

The main feedback and comments are paraphrased and summarized below.

- Tip
  - "What is PageRank"?
  - Missed both legend and question mark buttons.
  - "Put a delay on highlighting of the lines."
  - Not clear what the reputation means.
  - Users ask for explanation about meaning of circle text.
- Top
  - Network structure clear.
  - Circles are users/computers.
  - Lines are connections.
  - Notion there is some kind of reputation.
  - Beautiful animation, fun to use.

Adjustments

- Merge legend and question mark button into 'Help' button showing a help page.
- Replace PageRank reputation metric by easy to understand Balance metric.
- Delay on highlighting lines when hovering over lines.
- Use easy to understand and non technical naming.

A.2.2 Feedback gathered in the second round

The main feedback and comments are paraphrased and summarized below.

- Tip
  - Not clear which edge is highlighted when edges are overlapping.

  - "Why do I still see free riders? Are they blocked already?"
  - User tries right mouse click, this shows an unintended menu.
  - "How much can I download before I am blocked?"
  - "'Back To You' is animated in steps?"
  - Animation too fast, too much circles, hard to follow.

- Top
  - Network structure clear.
  - Balance and how to influence this is clear.
  - 'You' circle distinguishable from other circles.
  - 'Help' and 'Back To You' button found and clicked by most users.
  - 'Back To You' animation looks awesome.
  - "Fancy animation."

Adjustments

- Limit the amount of circles (and thereby the amount of edges) shown.
- Add text quantifying when users will be blocked.
- Rename 'Trust Display' to 'Network Explorer'.