Bilkent University

Department of Computer Engineering

Senior Design Project

High Level Design Report

Project Name: PeerNews

Group Members: Şamil İraz
Enes Emre Erdem
Muhammed İkbal Kazar

Supervisor: Prof. Özcan Öztürk

Jury Members: Prof. H. Altay Guvenir
Assoc. Prof. Selim Aksoy

Innovation Expert: Dr. Ali C. Beğen

High Level Design Report

May 17, 2019
This report is submitted to the Department of Computer Engineering of Bilkent University in partial fulfillment of the requirements of the Senior Design Project course CS491/2.

Table of Contents

1. Introduction

Decentralized architectures are becoming more popular due to recent improvements in peer to peer networking technologies. Crypto currencies such as Bitcoin, Ethereum and the file sharing systems such as Bittorrent are popular examples of renowned decentralized systems. PeerNews will provide a decentralized approach to media distribution by creating a peer to peer network among users. This decentralized architecture will not have a single point of failure or central authority. Thus everyone's freedom of speech rights will be protected by design.

1.1 Purpose of the System

PeerNews targets everyone who wants to participate in decentralized social media based on freedom of speech and privacy. The main purpose of the system is to let users share and consume messages of various mediums such as image, video or plain text in a secure, efficient and decentralized way. When consuming content, users able to specify their preferences to interact with the messages they are interested in.

1.2 Design Goals

1.2.1 Usability

- Interface will be user-friendly so that non-technical users are also encouraged to participate in the peer to peer network.
- Since the application itself is open-source, anyone who wants to create a better interface for their usage, they can change the visual parts as they wish.
- Users will be able to interact with the client application in a fast, simple and easy manner.

1.2.2 Scalability

- Since PeerNews does not have a centralized server, there cannot be an overload of users or messages. The performance issues are all related to the performance of WebTorrent.

1.2.3 Extensibility

- Users will be able to modify their client applications since it will be open source. This will allow for customizations on the client side.

1.2.4 Performance

- Latency of broadcasting a message should be less than 10 minutes in a crowded network of over 1 million nodes. Furthermore latency will be asymptotically bounded by O(logN) where N is the number of nodes in the network.

1.2.5 Reliability

- Application will be running all the time. Users will always have access to PeerNews.
- Messages will be accessible as long as there is at least one online node with that message since there is no single point of failure. Crash of one node does not harm any other node in the peer to peer network significantly.
- For the distribution of the actual content of the messages, existing torrent network will be used. Since torrent is an already established peer to peer network message-content delivery will be reliable.
- Authentication will be provided with the private-public key technology. As a result, there will not be any profile stealing.
- In the application, hashcash security system will be used so as to block any possible spam attacks to peers through network [3], [4].

1.3 Definitions, acronyms, and abbreviations

Open-source: source code that is freely available to everyone.
Message: Serialized message object that could be a post, comment, vote etc.
Peer: Client applications running on desktop.
Connector: Servers that help establish connection between peers through hole punching.
Feed: The list of all posts filtered according to the user-topic subscriptions.
Public Key: Public id that is shared with other users. Used to verify users.
Private Key: Private id that is not shared with other users. Used to sign posts.

1.4 Overview

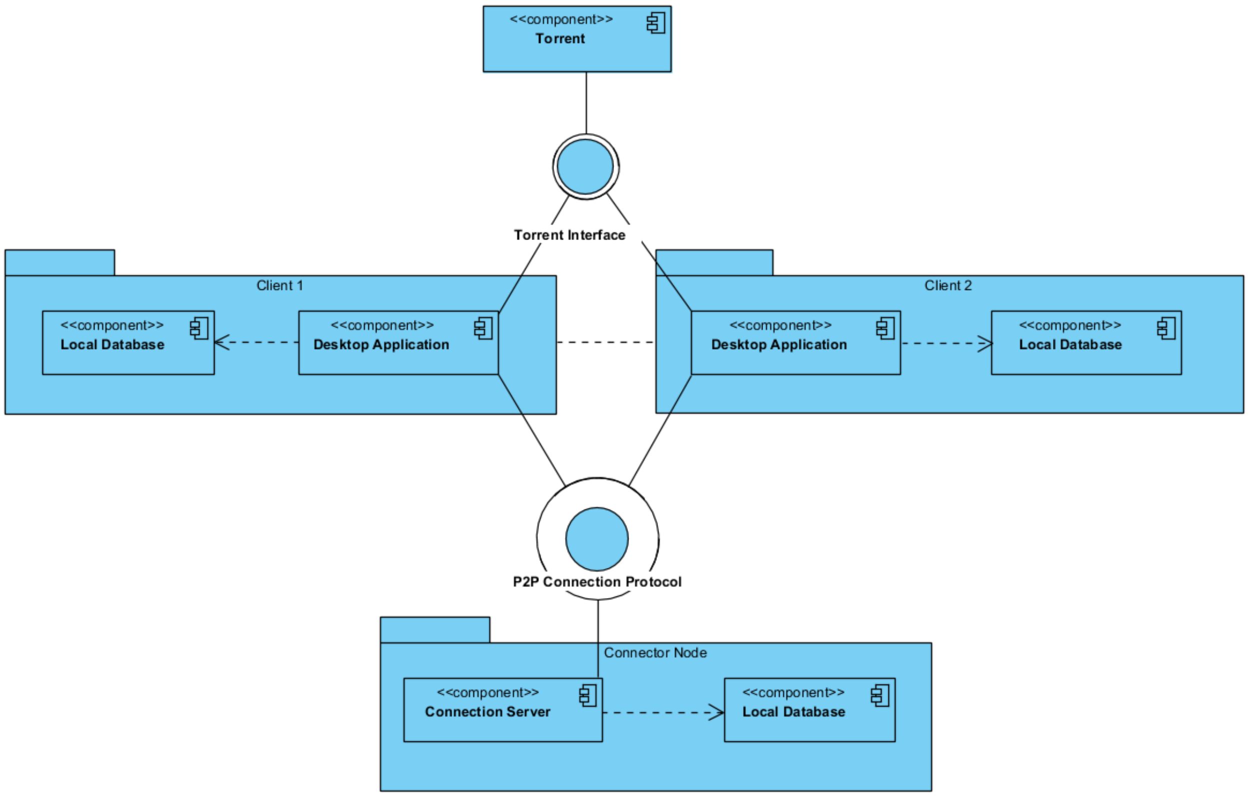
PeerNews consists of a p2p networking protocol and a client application. Messages are used to create a torrent file. Hash of the torrent and several other attributes are distributed through our own p2p network. Then we make use of the torrent network to seed and download the actual content of the message which are significantly larger in size. Client application downloads the content of messages from the torrent network according to user preferences. Client application receives messages from the p2p network and display them to the user appropriately.

2. Proposed Software Architecture

2.1 Overview

In this part of the report, the software architecture that we are planning to use is described in detail. First, subsystem decomposition of our software architecture is explained. The detailed information about persistent data management, access control, security and boundary conditions are discussed.

2.2. Subsystem Decomposition



2.2.1 Client

Client is user interaction component of the system, consisted of two portions: Desktop Application and Local Database.

2.2.1.1 Desktop Application

Desktop Application is the user interaction component of client. Simply all users use PeerNews through this application layer. The communication between people (other clients) is made through connections between applications. Application retrieves all of its required information to be shown to the user from the local database.

2.2.1.2 Local Database

Second portion of the Client component is the local database in which all of the local information of the user is held. Besides, ip addresses, public keys, recent posts, upvotes, comments etc. are kept in a local database to make searching faster and easier.

2.2.2 Connector Node

Connector nodes are responsible from connecting peers to each other. Connector nodes are consisted of two main parts. First one is the connection server, which actually creates the connection between peers. Second part is the local database of connector nodes.

2.2.2.1 Connection Server

Connection server will maintain a list of online peers (clients) along with their metadata regarding connection requirements. This will allow connection servers to let new clients discover other peers and connect to the network. Connection server will allow for the initial metadata message exchange to happen which will result in a true peer to peer connection between two clients. Once the peer to peer connection is established, connection server will disconnect from both clients. Connection Server will retrieve the required information to create connections from its local database.

2.2.2.2 Local Database

Connector nodes have all the information of the network in their databases. Public ip addresses of other connector nodes, users are stored in an efficient indexed manner so that searching and updating is faster.

2.2.3 Torrent

Torrent subsystem will be an abstraction to interact with the torrent network using an open source library that implements the protocol. This subsystem will allow peers (i.e. clients) to create new torrent files for new posts. Peers will also seed and download torrents for corresponding posts they have viewed by making use of this subsystem.

2.3. Persistent Data Management

Although there are no servers in our software architecture, peers will be responsible for maintaining and persisting data on disk for durability and efficient bandwidth usage. In particular, posts received by a peer will be stored until their expiration date in order to display them to the user and also seed posts back to the other peers that might need them in the future. In addition posts, other messages such as comments, posts etc. will also need to be persisted for similar reasons. Therefore all of the protocol layer messages will be persisted by the client desktop application. For this durable storage, we will be using SQLite since it is the most suitable database solution for desktop application due to its availability and compatibility with most operating systems.

2.4. Access Control and Security

PeerNews utilizes public-private key pair technology to provide decentralized authentication and security. For a new user, client application creates a key pair while keeping the private key secure [1], [2]. Then messages are cryptographically signed with private key. This signature can be verified by everyone in the network using public key. In addition messages are required to contain proof of work to prevent spam [3]. Proof of work guarantees that a set amount of computation has been done to send the corresponding message. Proof of work hash of the message can be verified instantly without requiring any burden on the verifying side. Hence peers are able to discard messages not containing the proper proof of work.

2.5. Boundary Conditions

2.5.1. Initialization

Users will need to go through our new user experience procedure involving key pair creation and initial topic subscriptions. Users will also need to be reminded that in the case of losing their private key, they will lose access to their account without any option of recovery. Client desktop application will require internet connection. It will also consume more bandwidth compared to average desktop application due to peer to peer nature of the system's upload requirement.

2.5.2. User and Data Termination

Users won't be able to close down their accounts or take down their prior posts. Account termination also won't be possible. However due to temporary nature of the messages with set expiration date, inactive users will essentially be removed from the network until their next activity.

2.5.3. Termination

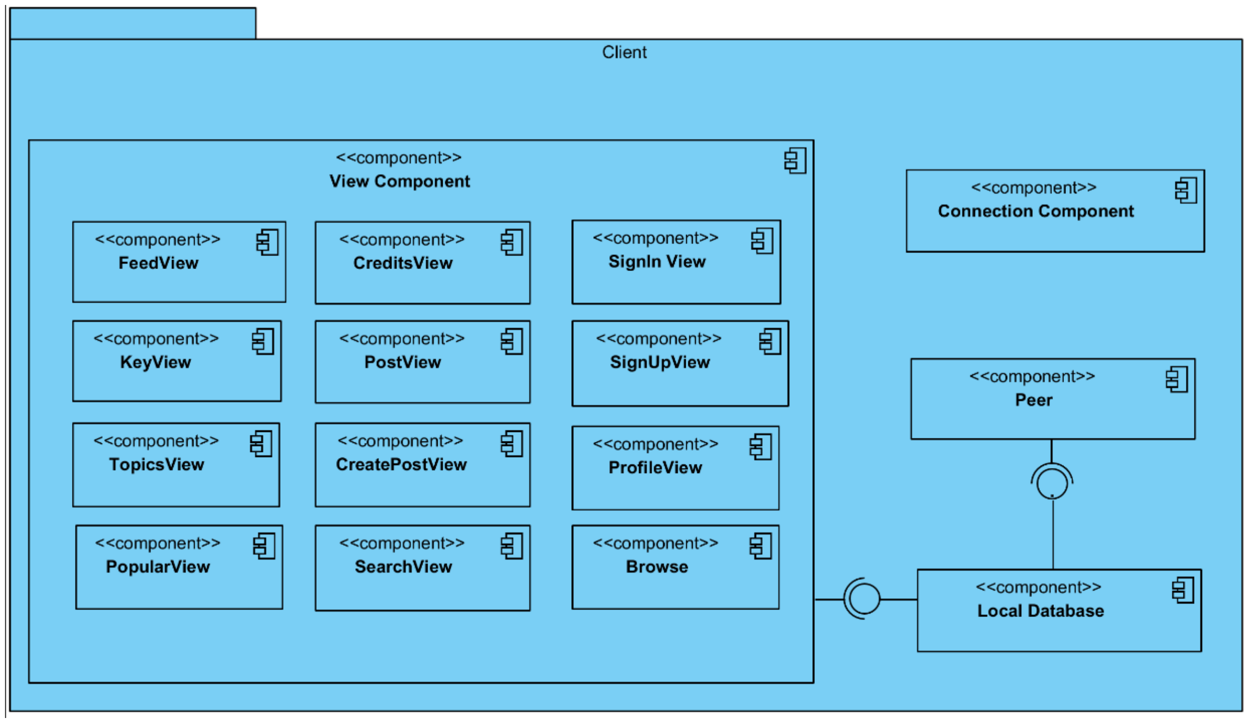
In the case of desktop application termination, posts of the users will be still broadcasted by other peers to the general network. In addition, messages that were sent after the desktop application's termination will be received by another peer when user's client becomes online again. Therefore, messages will not be lost due to application termination.

2.5.4. Dependence on Torrent Network

In order to consume media posts, users will need to download the media content of messages via the existing torrent network that is proven to be capable of handling large sizes files. However this also implies that users will be required to have access to the Torrent network. Some service providers and subnetworks might not allow access to the Torrent network as a result this will result in inaccessibility for PeerNews as well.

3. Subsystem Services

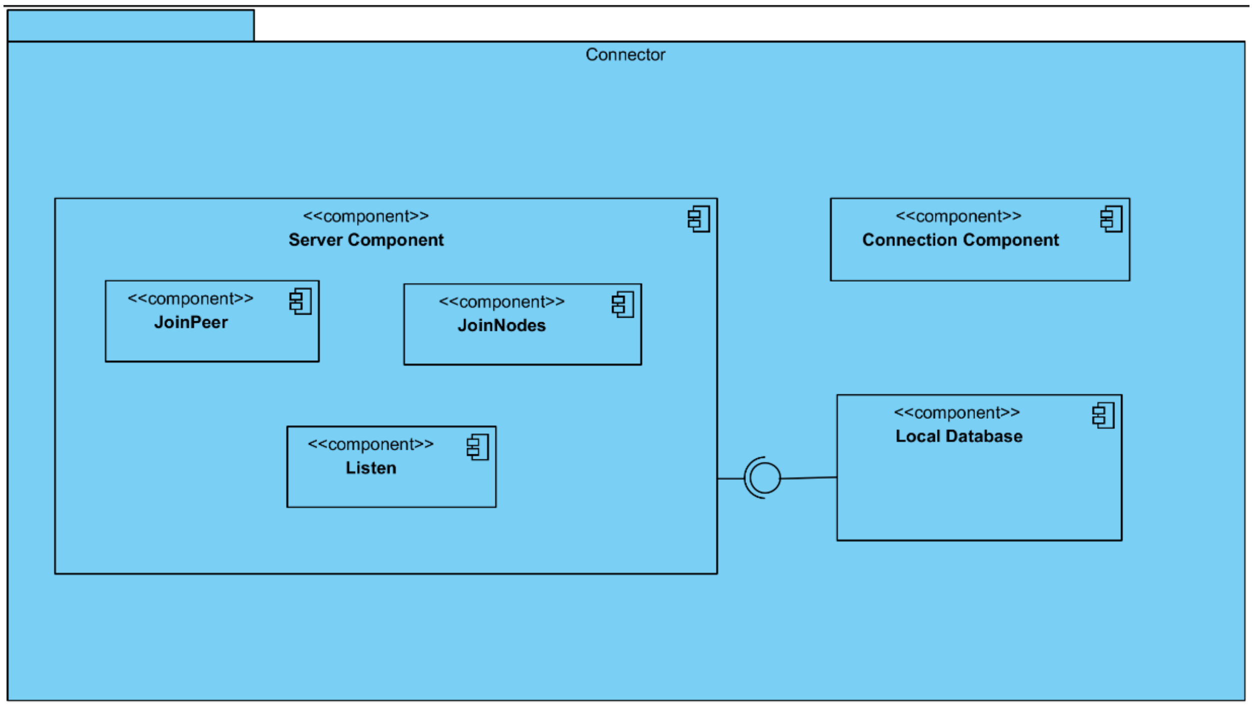
3.1. Client UI



- FeedView component will maintain and display the list of posts provided by protocol layer.
- PopularView component will be a variation of FeedView that displays trending posts.
- SearchView component will allow text based search on posts from subscribed topics.
- ProfileView will be responsible for displaying user information, settings and misc. Functionality such as exporting keys.
- CreatePostView component and related subcomponents will handle post creation UI.

- PostView component will display a given post and provide interactivity required to perform several user actions such as commenting and voting.
- SignInView component will display new user experience related pages.
- SignUpView component will handle the process of creating a new user account and subscribing to first topics to create a new user feed.
- KeyView component will be responsible for displaying the public and private key of the user.
- TopicsView component will display a list of topics and allow users to subscribe to topics.
- CreditsView component will display information about open source developers behind the system's development.
- Browse component will let users browse their local file system to upload content.

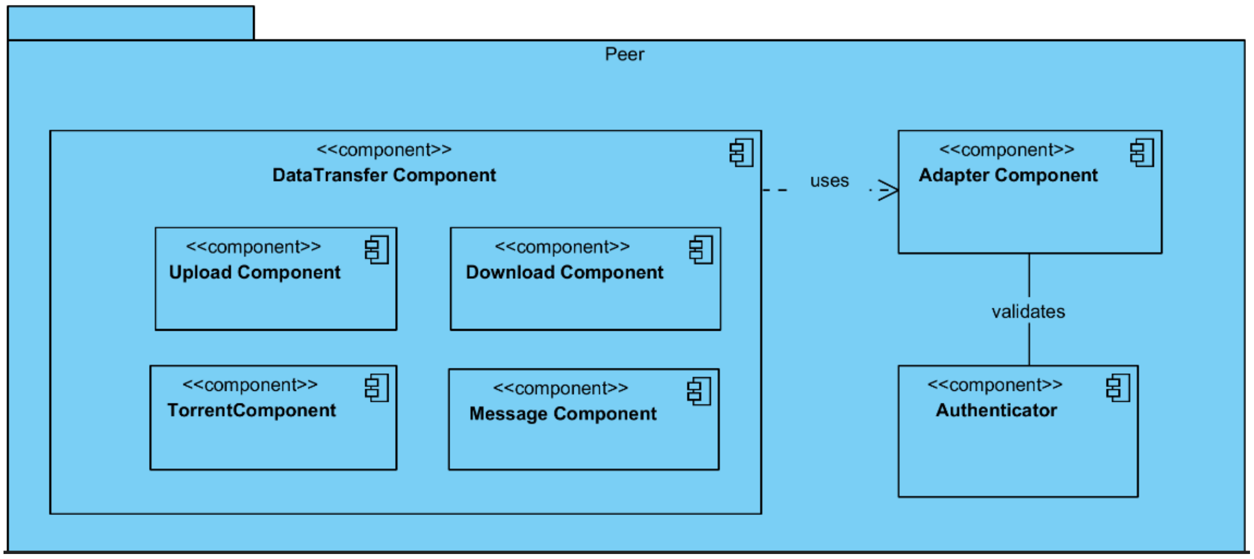
3.2. Connector



Since our system is designed to be decentralized and peer to peer, Connector nodes will play a minor role and thus they will be very simple. Connector nodes will only be responsible for establishing the initial connection and discovery phase between two peers.

- Server component maintains a list of peers that are known to be online at the moment by the connector node by checking the local information of peers stored in the local database.
- JoinPeer component will retrieve the required information about an online peer to be able to establish a peer to peer connection through TCP hole punching.
- JoinNode component will take the information about other connector nodes known by this particular connector node from the local database. This component compares that information with the information that is stored in the Connector Nodes, so that overall network will not be partitioned where peers connected through different connector nodes are invisible to each other.
- Connection component sets up the real-time connections to peers and connector nodes.

3.3. Peer



- Message component is the component that handles all kinds of message objects. Each message will contain various meta-data such as signature, hash, proof of work.
- Request, Comment, Vote, Post are parts of the Message component. Each represents a particular type of message with additional fields.

- Authenticator component will be responsible for creating a digital signature and proof of work for a given message.
- KeyPair class contains private and public keys of the user. It will also be responsible for verification and initial generation of keys. This class is also a part of the Authenticator component.
- Adapter component maintains a list of connectors and peers that are directly connected to the current user. All of the messages received by Adaptor will also be stored in SQLite before they are processed or deserialized for persistency. Upload and download components are responsible from uploading hash codes to peers and downloading hash codes from peers.
- Torrent component will be responsible for the connection with the torrent network. Torrent component also handles seeding of the messages.

4. References

[1] “Managing Trust in Peer-to-Peer Systems.” *10 Best Practices for Secure Software Development | Security, Data and Privacy | Subject Areas | Publishing and Editorial | BCS - The Chartered Institute for IT*, ITNOWextra, Jan. 2006, www.bcs.org/content/conWebDoc/3059.

[2] Kamvar, Sepandar D., et al. “The Eigentrust Algorithm for Reputation Management in P2P Networks.” *Contents: Using the Digital Library*, ACM, 20 May 2003, dl.acm.org/citation.cfm?id=775242.

[3] Adam Back, "Hashcash - A Denial of Service Counter-Measure", technical report, August 2002

[4] Object-Oriented Software Engineering, Using UML, Patterns, and Java, 2nd Edition, by Bernd Bruegge and Allen H. Dutoit, Prentice-Hall, 2004, ISBN: 0-13-047110-0.