**Project Specification**

**Title**

Peer to Peer ChatSDk with distributed hash tables

**Student(s)**

Benedikt Kaiser BSc.

**Project Advisor**

Erik Sonnleitner FH-Prof. DI Dr.

**Date and Time**

Hagenberg, 27.06.2023

Version: 03/2023

Table of Contents

[1. Overall Goal and Motivation 3](#_Toc139628332)

[2. Project Description 3](#_Toc139628333)

[2.1 Initial Technical Situation 3](#_Toc139628334)

[Currently, various distributed products, such as Matrix, are available. However, these products require you to set up a server at home with the necessary software to host the chat. While this setup offers security, it comes with the significant drawback of requiring a server at home and having the necessary knowledge. 3](#_Toc139628335)

[2.2 Realisation Concept 3](#_Toc139628336)

[2.3 System Context 4](#_Toc139628337)

[2.4 User Interfaces 4](#_Toc139628338)

[2.5 Functionality 4](#_Toc139628339)

[2.6 Technical Data 5](#_Toc139628340)

[3. Development Conditions 5](#_Toc139628341)

[3.1 Software 5](#_Toc139628342)

[3.2 Hardware 5](#_Toc139628343)

[4. Impact 5](#_Toc139628344)

[4.1 Target Groups 5](#_Toc139628345)

[4.2 Areas of Application 5](#_Toc139628346)

[4.3 Estimation of the Market Chances 5](#_Toc139628347)

[A. Appendix 5](#_Toc139628348)

# Overall Goal and Motivation

### Basic Idea

The fundamental concept underlying my intended work is the development of a Distributed ChatSDK for iOS devices utilizing the LibP2P framework. This endeavor is motivated by the increasing demand for decentralized communication platforms that prioritize privacy, security, and resilience against centralized control.

### Motivation:

**Privacy and Security in Strict Countries**: Decentralized communication systems offer secure and private communication channels for individuals in countries with stringent internet regulations, bypassing centralized servers that can be subject to surveillance and censorship.

**User Empowerment**: Decentralized communication systems empower users by providing greater control over their personal data, mitigating concerns about data ownership and privacy that arise in centralized architectures.

**Network Resilience**: Decentralized systems demonstrate resilience by adapting to network disruptions and maintaining connectivity through alternative routes, ensuring uninterrupted communication even when parts of the network are compromised or unavailable.

**Trust and Verification**: Trust and verification mechanisms are crucial components of decentralized communication systems, enabling secure interactions and fostering confidence among users through cryptographic techniques, consensus algorithms, and robust trust models.

# Project Description

## Initial Technical Situation

Currently, various distributed products, such as Matrix, are available. However, these products require you to set up a server at home with the necessary software to host the chat. While this setup offers security, it comes with the significant drawback of requiring a server at home and having the necessary knowledge.

### Kademlia

Maymounkov and Mazieres write in their paper about using a XOR-metric to improve DHTs. Kademlia minimizes the number of configuration messages by using the key lookups to also learn about other nodes The basic setup is like any other basic DHT: Keys are opaque, 160-bit quantities. Every computer has a unique node ID in the 160 bit key space. Key-Value pairs are stored on nodes with the ID “close“ 2 to the key. According to the authors, the main advantage of Kademlia is using the XOR-metric to calculate the distance between two nodes.Every node has a routing table which is a small part of the whole routing table. In this routing table a triples with 〈IP address, UDP port, Node ID〉gets stored. We call this routing table within a node K-Bucket. The idea behind the paper is, storing one address of each node in the K-Bucket that is in every subtree that does not contain our initial node. Due to the fact that there is a node from every sub tree stored in our K-Bucket, it gets checked which Node ID in our K-Bucket is closest to the Node ID we want to communicate to. In Figure 4 the black dot (our node) wants to communicate to 1110. Which means a first message gets send to the only address from that sub tree that is stored in our routing table.

### Network address translation (NAT) and firewalls

After using Kademlia to retrieve the public IP address of our communication partner, it is most of the time still not possible to create a connection due to Firewalls and NAT [8].

According to [8] 52% of the nodes of the InterPlanetary File System (IPFS), a distributed peer-to-peer network, are using NAT and thus not publicly reachable. Technologies like WebRTC have the same problem but use servers to solve it. In their paper [8], the authors try solving the challenge of connecting to a private address behind a firewall without additional server. They use the hole punching method.

### LibP2P

LibP2P is a modular Peer-to-Peer network stack. LibP2P was originally developed as a base for the IPFS but has phased out as a network stack [1]. Already a lot of different P2P applications use LibP2P as their network stack. According to the official LibP2P website, developers use the library because of the following reasons:

• Modularity

• Versatility

• Extensive transport configurability

• Piercing NAT Barriers.

## Realisation Concept

The proof-of-concept implementation for our distributed ChatSDK on iOS, utilizing LibP2P, realizes several key functionalities essential for decentralized communication within a local network. These functionalities include:

1. **Chat Functionality:** The primary feature of the ChatSDK is to enable users to engage in real-time messaging within a local network environment. This includes sending and receiving text messages, multimedia content, and other forms of data supported by the application.
2. **Network Address Translation (NAT) Handling:** The implementation incorporates mechanisms to handle NAT traversal, allowing devices behind NAT devices to establish direct communication channels. This feature is crucial for enabling peer-to-peer communication within private networks where NAT is commonly encountered.
3. **Peer Discovery:** The ChatSDK implements peer discovery mechanisms to locate and connect with other devices running the application within the same local network. This involves identifying available peers and establishing connections to facilitate messaging.

## System Context

The intended artefacts collaborate with external components and interfaces to fulfil their functionalities:

* LibP2P Library: The ChatSDK leverages the LibP2P library to implement peer-to-peer communication capabilities, including network abstraction, peer discovery, and secure messaging.

## User Interfaces

The actual master project implementation will not have a user interface but for testing purposes there will be an example implementation with a primitive user interface. This will mainly consist of necessities and will not implement eye candy.

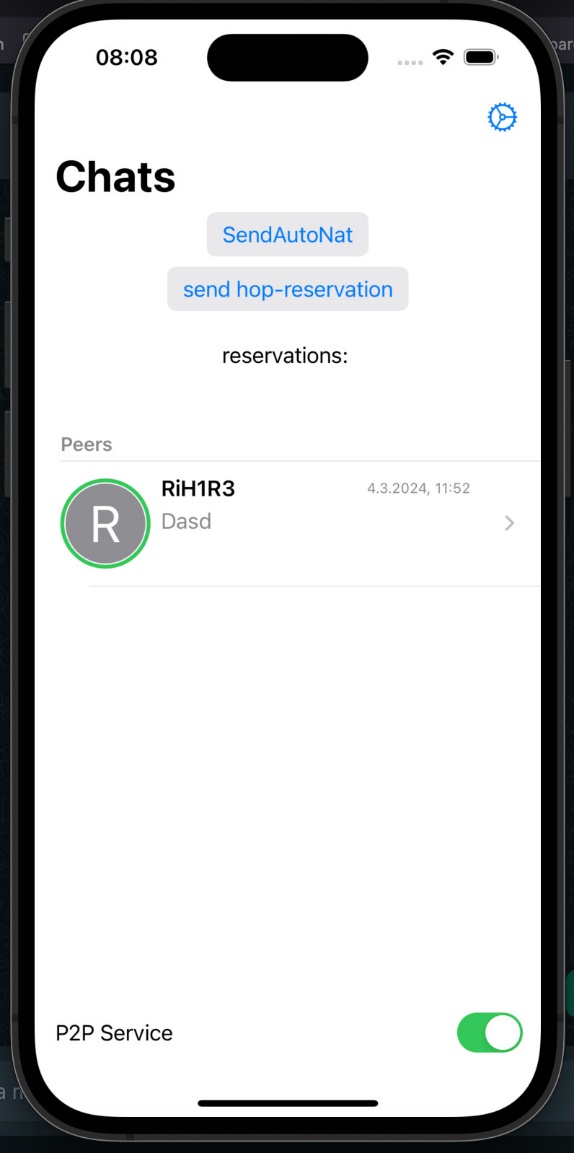


Figure 1 user interface

The UI for this example app can be seen in Figure 1.

Functionality

In this screen it is possible to see the chat window and if another user is active. All the way on the top It is possible to see two buttons. The SendAutoNat button sends an Autonat request to nearby nodes to determine whether the node is publicly reachable or not. If the node is not publicly reachable, the node can make a reservation at a publicly reachable node. So, every future connection with other nodes is going to be initiated via this reservation.

The messages are currently only stored in the devices storage.

It is also possible to activate and deactivate the P2P Service with the switch at the bottom of the screen.

If you press on the peer you open up the chat and it is possible to chat with your peer. In the settings it is possible to change your nickname and to reset all the stored messages.

# Development Conditions

## Software

XCode

## Hardware

MacBook, iPhone

# Impact

## Target Groups

Developers: Primary stakeholders include developers and software engineers interested in building decentralized communication applications for iOS platforms. The ChatSDK provides a comprehensive framework and set of tools for integrating decentralized messaging functionalities into their own applications, reducing development time and complexity.

Research Community: Researchers and academics in the fields of distributed systems, peer-to-peer networks, cryptography, and decentralized technologies constitute a target group interested in exploring the technical aspects and implications of the ChatSDK. The project may serve as a case study or basis for further research in decentralized communication systems.

## Areas of Application

Ios App development

## Estimation of the Market Chances

This project is going to be an open-source project. There is no point in making a closed source project and just say that it is secure. The developer needs to be able to check it if they want to.

# Appendix

Kademlia:

<https://pdos.csail.mit.edu/~petar/papers/maymounkov-kademlia-lncs.pdf>

<https://libp2p.io/>