**Emulating custom TCP (acknowledge based) protocol using UDP**

High Level Design & Low Level Design

The purpose of this document is to provide a template for documenting both HLD & LLD.

**Document Control :**

| **Project Revision History** | | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | |  |  |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | | **Approver Signature** | |
| 27/08/2022 | Version 0.1 | Praveenram S | Changes in handshake function and send data functionality | | | |  | |
| 27/08/2022 | Version 0.2 | Neha Kumari | Changes in handshake function and display data functionality | | | |  | |
| 27/08/2022 | Version 0.3 | Roshni Singha Prajapati | Changes in login and validation function | | | |  | |
| 27/08/2022 | Version 0.4 | Saloni Naikwadi | Changes in coding guidelines | | | |  | |
| 27/08/2022 | Version 0.5 | Shivani Mahale | Changes in naming convention | | | |  | |

[**1. Introduction 3**](#_heading=h.3znysh7)

[1.1. Intended Audience 3](#_heading=h.2et92p0)

[1.2. Acronyms/Abbreviations 3](#_heading=h.tyjcwt)

[1.3. Project Purpose 3](#_heading=h.3dy6vkm)

[1.4. Key Project Objectives 3](#_heading=h.1t3h5sf)

[1.5. Project Scope and Limitation 3](#_heading=h.4d34og8)

[*1.5.1. In Scope 3*](#_heading=h.2s8eyo1)

[*1.5.2. Out of scope 3*](#_heading=h.17dp8vu)

[1.6. Functional Overview 3](#_heading=h.3rdcrjn)

[1.7. Assumptions, Dependencies & Constraints 3](#_heading=h.26in1rg)

[1.8. Risks 3](#_heading=h.lnxbz9)

[**2. Design Overview 3**](#_heading=h.35nkun2)

[2.1. Design Objectives 3](#_heading=h.1ksv4uv)

[*2.1.1. Recommended Architecture 3*](#_heading=h.44sinio)

[2.2. Architectural Strategies 3](#_heading=h.2jxsxqh)

[*2.2.1. Design Alternative 3*](#_heading=h.z337ya)

[*2.2.2. Reuse of Existing Common Services/Utilities 3*](#_heading=h.3j2qqm3)

[*2.2.3. Creation of New Common Services/Utilities 3*](#_heading=h.1y810tw)

[*2.2.4. User Interface Paradigms 3*](#_heading=h.4i7ojhp)

[*2.2.5. System Interface Paradigms 3*](#_heading=h.2xcytpi)

[*2.2.6. Error Detection / Exceptional Handling 3*](#_heading=h.1ci93xb)

[*2.2.7. Memory Management 3*](#_heading=h.3whwml4)

[*2.2.8. Performance 3*](#_heading=h.2bn6wsx)

[*2.2.9. Security 3*](#_heading=h.qsh70q)

[*2.2.10. Concurrency and Synchronization 3*](#_heading=h.3as4poj)

[**3. System Architecture 3**](#_heading=h.49x2ik5)

[3.1. System Architecture Diagram. (Not Necessary) 3](#_heading=h.2p2csry)

[3.2. System Use-Cases 3](#_heading=h.147n2zr)

[**4. Detailed System Design 3**](#_heading=h.1hmsyys)

[4.1. Key Entities](#_heading=h.41mghml)

4.2 data conversion [3](#_heading=h.41mghml)

[4.3. Disaster and Failure Recovery 3](#_heading=h.4f1mdlm)

[4.4. Variables 3](#_heading=h.28h4qwu)

[4.5. Data Migration 3](#_heading=h.1mrcu09)

[*4.5.1. Architectural Representation 3*](#_heading=h.46r0co2)

[*4.5.2. Logical View 3*](#_heading=h.111kx3o)

[*4.5.3. Data model 3*](#_heading=h.206ipza)

[*4.5.4. Deployment View 3*](#_heading=h.3ygebqi)

[**5. Environment Description 3**](#_heading=h.2dlolyb)

[5.1. Time Zone Support 3](#_heading=h.sqyw64)

[5.2. Language Support 3](#_heading=h.3cqmetx)

[5.3. Server/User Desktop Requirements 3](#_heading=h.1rvwp1q)

[*5.4.1. Deployment Considerations 3*](#_heading=h.2r0uhxc)

[*5.4.2. Application Server Disk Space 3*](#_heading=h.1664s55)

[*5.4.3. Database Server Disk Space 3*](#_heading=h.3q5sasy)

[*5.4.4. Integration Requirements 3*](#_heading=h.25b2l0r)

[*5.4.5. Jobs 3*](#_heading=h.kgcv8k)

[*5.4.6. Network 3*](#_heading=h.34g0dwd)

[*5.4.7. Others 3*](#_heading=h.1jlao46)

[5.5. Configuration 3](#_heading=h.43ky6rz)

[*5.5.1. Operating System 3*](#_heading=h.2iq8gzs)

[*5.5.2. Database 3*](#_heading=h.xvir7l)

[*5.5.3. Network 3*](#_heading=h.3hv69ve)

[*5.5.4. Desktop 3*](#_heading=h.1x0gk37)

[**6. References 3**](#_heading=h.4h042r0)

[**7. Appendix 3**](#_heading=h.2w5ecyt)

# 

# Introduction

**Transmission Control Protocol:**

TCP stands for Transmission Control Protocol, a communications standard that enables application programs and computing devices to exchange messages over a network. It is designed to send packets across the internet and ensure the successful delivery of data and messages over networks.

**User Datagram Protocol:**

UDP (User Datagram Protocol) is a long-standing protocol used together with IP for sending data when transmission speed and efficiency matter more than security and reliability. UDP uses a simple connectionless communication model with a minimum protocol mechanism.

UDP provides checksums for data integrity, and port numbers for addressing different functions at the source and destination of the datagram. It has no handshaking dialogues, and thus exposes the user's program to any unreliability of the underlying network; there is no guarantee of delivery, ordering, or duplicate protection. If error-correction facilities are needed at the network interface level, an application may use the Transmission Control Protocol (TCP).

## Intended Audience

| BU from Capgemini |  |
| --- | --- |
|  |  |

## Acronyms/Abbreviations

| UDP | User Datagram Protocol |
| --- | --- |
| TCP | Transmission Control Protocol |

## Project Purpose

The purpose of this project is to implement a TCP protocol using UDP transmission. TCP 3-way handshake or the send/receive methods should be implemented. In this regard, we will be implementing TCP over UDP Socket.

## Key Project Objectives

The objective of the project is to create an enhanced UDP .

### In Scope

This project is tested for study purposes.

### Out of scope

This project is tested for demonstration purposes.

## Functional Overview

1. Client:

In the client terminal the client will request the server using the query like the IP address or the hostname of the computer that DNS server is running on, port is the port the DNS server is listening on, and hostname is the domain name the user is trying to resolve into an IP address.

For example:

. /client 8787

1. Server:

In the server terminal the server will receive the request from the client. In the response to that request the server will first read from the text file which includes the IP address and domain name and then it will compare the domain name/hostname with the hostname provided by the client and accordingly will send the response with the IP address and the message if that domain name is active/Local or not to the client.

For example:

. /server 8787

## Assumptions, Dependencies & Constraints

OPERATING SYSTEMS:

Operating environment for implementing DNS are:

* Client/server system
* Operating system: Linux
* Platform: Ubuntu/C

## Risks

As it is a study purpose project so it doesn’t have any risk.

# Design Overview

Like TCP, ToU has a client and a server. A client connects to a TCP server to establish a ToU connection. Below, we describe the key ToU operations.

Like TCP, ToU uses a three-way handshake to establish a connection. Similarly, it follows TCP's semantics in tearing down the connection.

## Design Objectives

UDP header is an 8-bytes fixed and simple header, while for TCP it may vary from 20 bytes to 60 bytes. The first 8 Bytes contain all necessary header information and the remaining part consists of data. UDP port number fields are each 16 bits long, therefore the range for port numbers is defined from 0 to 65535; port number 0 is reserved. Port numbers help to distinguish different user requests or processes.

### Recommended Architecture

TCP enables applications to communicate with each other as though connected by a physical circuit. TCP sends data in a form that appears to be transmitted in a character-by-character fashion, rather than as discrete packets. This transmission consists of a starting point, which opens the connection, the entire transmission in byte order, and an ending point, which closes the connection. TCP confirms that a packet has reached its destination by establishing an end-to-end connection between sending and receiving hosts. TCP is therefore considered a “reliable, connection-oriented” protocol.

## Architectural Strategies

The UDP provides a connectionless service for application-level procedures. UDP doesn't guarantee delivery, preservation of sequence, or protection against duplication. UDP enables a procedure to send messages to other procedures with a minimum of protocol mechanisms. Some transaction-oriented applications make use of UDP; one example is the *Simple Network Management Protocol*

### Design Alternative

UDP is used when you need a translation of a domain name. Client sends a question DNS server answers with IP(if it has it in its record). All done with simple Datagram packets. No worry if a packet is lost or corrupt, the client can ask for it again.

### 

### 

### Reuse of Existing Common Services/Utilities

#include<stdlib.h>

#include<stdio.h>

#include<string.h>

#include<unistd.h>

#include<sys/types.h>

#include<sys/socket.h>

#include<netinet/in.h>

### Creation of New Common Services/Utilities

We are using inbuilt Libraries. and also for client and server we are creating Header files.

### User Interface Paradigms

Command Line Interface: Terminal

### System Interface Paradigms

Command Line Interface: Terminal

### Error Detection / Exceptional Handling

Error detection :

1. Invalid ClientSocket
2. Invalid connect value.

Errors will be handled by Error Handling Condition

### Memory Management

When you create a deduplication data store, you specify whether the hash destination is on a Solid State Drive (SSD mode) or the hard disk drive (RAM mode). If you configured the hard disk as the hash destination, you need more memory to process hash keys. As a result when your backup size grows, all your memory may get exhausted. In that case, you can add an SSD to back up more data. Similarly if you had configured an SSD as the hash destination, you need less memory to process hash keys.

### Performance

The optimal settings of the tunable communications parameters vary with the type of LAN, as well as with the communications-I/O characteristics of the predominant system and application programs

### Security

When it comes to UDP and security, it all depends on the UDP service that is running on a port and how secure the service is. The service could be vulnerable to hacking if the service has an exploit or a bug in it that allows remote access, overflow, etc.

### Concurrency and Synchronization

Use the procedure detailed below to Concurrency and synchronize a restarted or added in a cluster, with the TCP or UDP protocol:

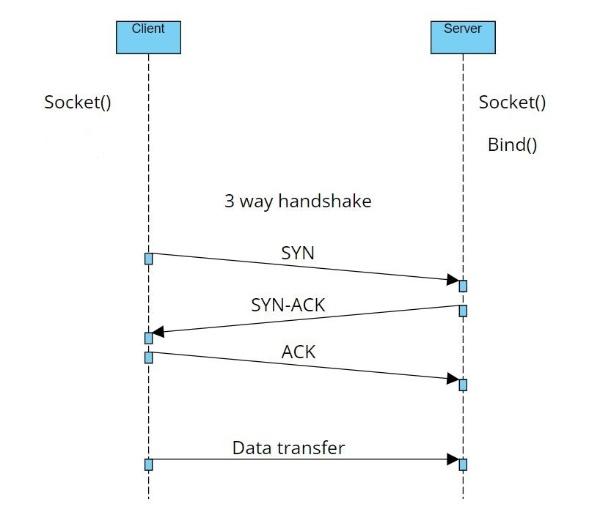
**TCP:** The recommended communication protocol to synchronize maps on start-up because it is reliable and efficient.  
 **UDP**: For fast communication, but has drawbacks. For instance, packets can get lost.

# System Architecture

The UDP provides a connectionless service for application-level procedures. UDP doesn't guarantee delivery, preservation of sequence, or protection against duplication. UDP enables a procedure to send messages to other procedures with a minimum of protocol mechanisms. Some transaction-oriented applications make use of UDP; one example is the *Simple Network Management Protocol*



## System Architecture Diagram. (Not Necessary)



## System Use-Cases

A use case can have multiple paths to reach the goal; each of them is considered a use case scenario. In simple words, a use case is a goal with various processes, and a case scenario represents a linear and straight path through one of the operations.

# Detailed System Design

The client has to register and then we are validating them.The client can login to the system.And once the login is successful a socket is created and the server gets binded to the client. After the clientAfter the client sends a string, we are storing the data in the client and we are sending the data character by character and acknowledging whether the data is sent or not and we are storing the data sequentially.,by performing a 3 way handshake we can prevent data loss.

## Key Entities

Creating socket

Binding the socket

Sending and receiving messages between client and server.

### Data Conversion

**atoi** - Character array is converted into integer value.

**htons -** Integer value is converted into Network readable form.

## Disaster and Failure Recovery

The three phases of disaster recovery include assessment, restoration, and recovery. When disaster strikes, you want to get back to normal as quickly as possible. It's important to go through these three phases of disaster recovery.

## Variables

Server part:

socketFileDescriptor, serverLength, clientLength, send, receive, buffor, argc, argv.

Client part:

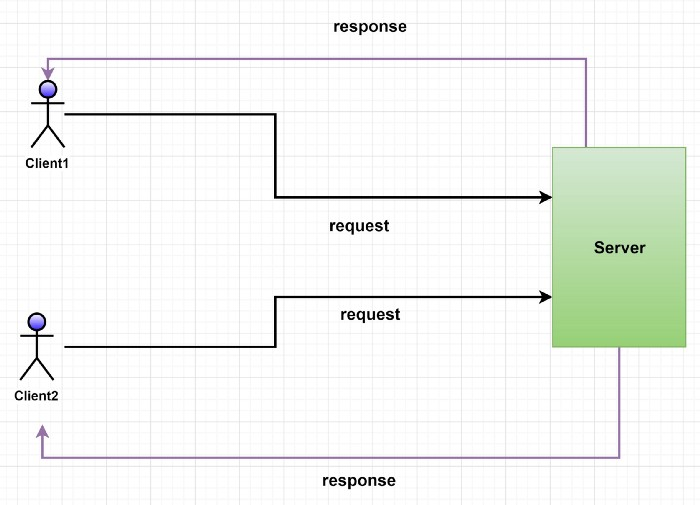
socketFileDescriptor, clientLength, send, receive, buffor, argc, argv.

## Data Migration

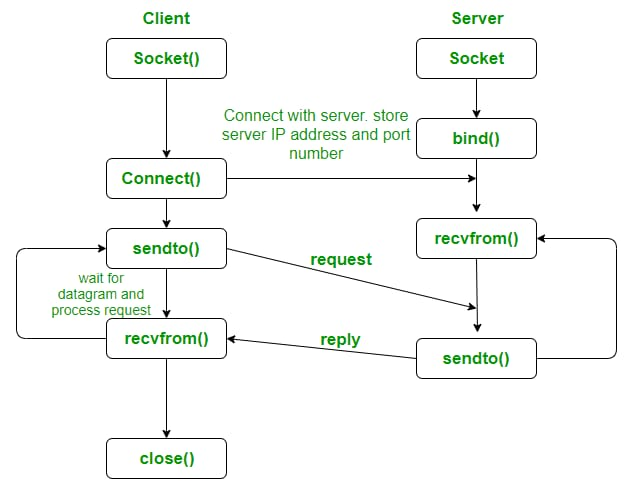
To make sure that each message reaches its target location intact, the TCP/IP model breaks down the data into small bundles and afterward reassembles the bundles into the original message on the opposite end. Sending the information in little bundles of information makes it simpler to maintain efficiency as opposed to sending everything in one go.

After a particular message is broken down into bundles, these bundles may travel along multiple routes if one route is jammed but the destination remains the same.

### Architectural Representation

**

### Logical View

**

### Data model

Legacy system data model

Proposed system data model

Interface data model

### Deployment View

NA

# Environment Description

**UBUNTU**: Ubuntu is an open-source operating system (OS) based on the Debian GNU/Linux distribution. Ubuntu incorporates all the features of a Unix OS with an added customizable GUI, which makes it popular in universities and research organizations. Ubuntu is primarily designed to be used on personal computers, although a server edition does also exist.

**C language**: C is a procedural programming language. It was initially developed by Dennis Ritchie in 1972. It was mainly developed as a system programming language to write an operating system. The main features of the C language include low-level memory access, a simple set of keywords, and a clean style. These features make C language suitable for system programming like an operating system or compiler development.

Many later languages have borrowed syntax/features directly or indirectly from the C language. Like the syntax of Java, PHP, JavaScript, and many other languages are mainly based on the C language.

## Time Zone Support

India Standard Time is the time zone observed throughout India, with a time offset of UTC+05:30.

## Language Support

C programming language

## Server/User Desktop Requirements

● CPU (circa 2013+): Intel i3/i5/i7 generation 3 and later

● RAM: 4GB or greater - For optimal performance, 6GB or 8GB are recommended if you will be running multiple browser tabs and/or multiple applications at the same time

● Internal memory:512 GB SSD/HDD.

### 

### Application Server Disk Space

15Gb space

### Database Server Disk Space

15Gb space

### Jobs

Interaction between client and server.

### Network

IPV4- Internet Protocol version 4 is the fourth version of the Internet Protocol. It is one of the core protocols of standards-based internetworking methods in the Internet and other packet-switched networks.

## Configuration

8Gb ram

Windows 10

SSD 500

### Operating System

Ubuntu

### Database

Files

### Network

IPV4

### Desktop

Using Ubuntu in VM in windows desktop

# References

1. <https://lifesize.com/en/blog/tcp-vs-udp/#:~:text=TCP%20is%20a%20connection-oriented,is%20only%20possible%20with%20TCP>
2. <https://github.com/kyclark/make-tutorial>
3. https://github.com/Azmah-Bad/TCPoverUDP

# Appendix

*[This section should provide a complete list of all documents or links on the Internet where related material can be found.]*

<https://github.com/Azmah-Bad/TCPoverUDP>

**Change Log**

| **QMS Template Version Control (Maintained by QA)** | | | | | |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
| **Date** | **Version** | **Author** | | **Description** | |
| 28-May-2015 | 1.0 | QA Team | | Initial Version | |
|  |  |  | |  | |
|  |  |  | |  | |
|  |  |  | |  | |