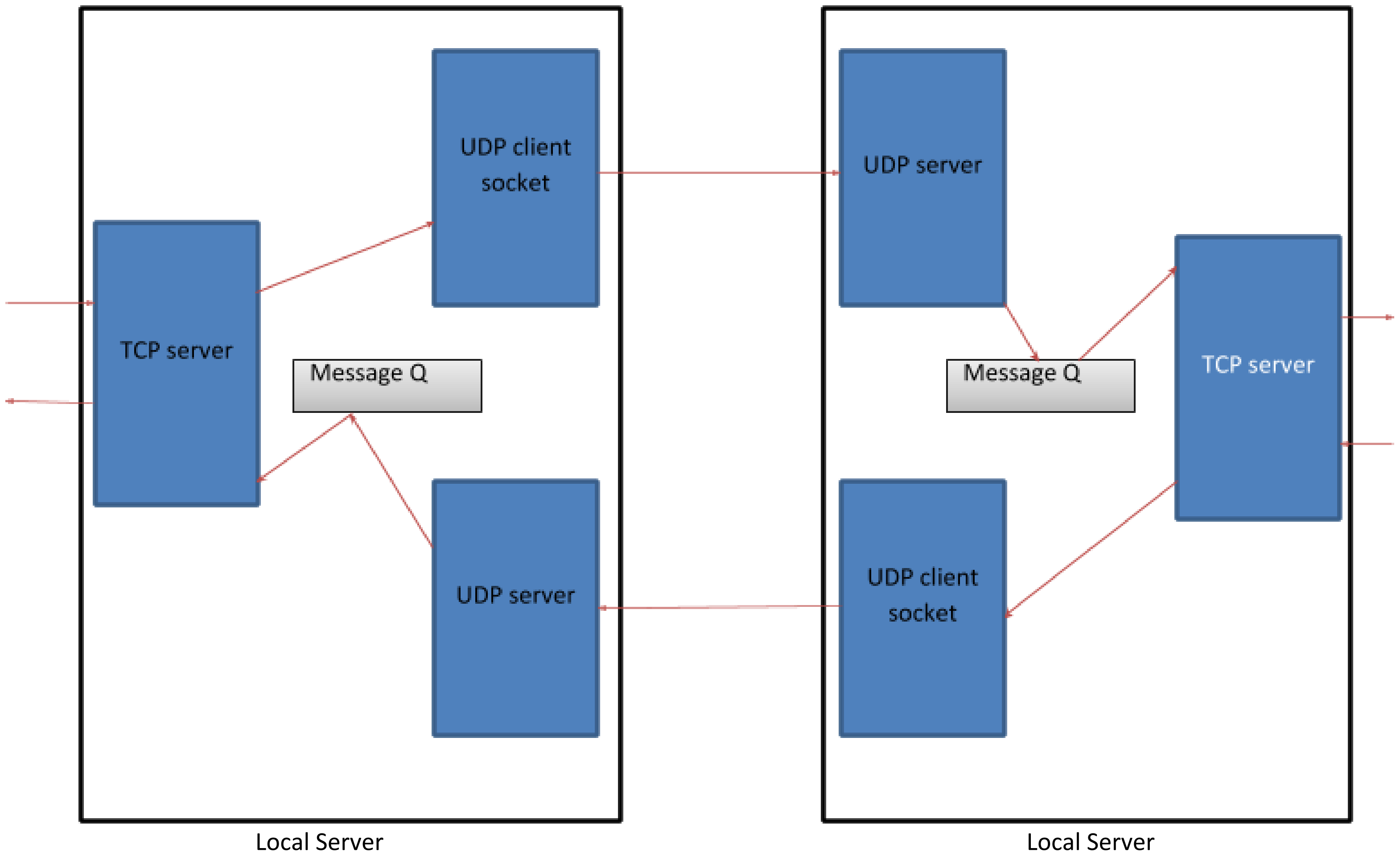
Denson George (2018H1030133P)

Network Message Bus

Block diagram:



Design Decisions:

Mtype field:

Long mtype field consisting of ip address and port number. IP address and port number both stored in Network Byte Order and the resulting long (6 Bytes) is also stored in Network Byte Order. This mtype field is also used as a type field for the locally saved Message Queues thus keeping a consistent design throughout.

2B(0's)	IP address (4Bytes)	2B(Port)

Message Queues:

This is primarily used as an IPC mechanism between the TCP server process and the UDP server process. Typically for offline processes, the UDP server keeps the messages stored in Message

Network Message Bus

Queue so that they can be retrieved once a process has bound to a port and has connected to the TCP server.

TCP server:

A multi process concurrent server so that multiple clients are handled concurrently. All processes irrespective of whether they want to send /receive a message contacts the TCP server. TCP server co-ordinates with the UDP server as shown above to send or receive messages. TCP server process uses UDP client socket to send messages to remote hosts in the network. An option field is also present as part of the message, this helps to distinguish send and receive requests at the TCP server side.

UDP server:

Receive message for my host and save it in the message queue. This can then be consumed immediately if the process is available or is stored in the queue and can be retrieved at a later time when the process connects to the TCP server with a receive request. A **separate process** waits on receive to get messages.

Message structure:

```
struct payload

{ int option; char data[1024]; };

struct msg { unsigned long mtype; struct payload pload; };

// the main structure that is passed along in the network
```

API design:

msgsnd_nmb(char[] destip, short destport,char[] message)

Send function that can be used by a process to send messages to a remote process specified by the ip address and port number specified in the parameters. This function performs the necessary conversions and encapsulates all the details into a structure and passes it onto the local TCP server running at 1111. This also sets option field as 1 indicating a sending event at the local TCP server.

Return type of the function is void.
In case of any errors the program exits and an error message with cause of the error is displayed on the screen.

msgrcv_nmb (char ip[], unsigned short port)

Receive function bounds the calling process to the end point specified in the parameters. Through the bound socket connection is created and any available message for the endpoint is fetched. This also sets option field as 2 indicating a sending event at the local TCP server. Port numbers less than and equal to 1023 are not allowed as they are reserved ports.This is also handled in the function.

Return type: struct msg

In case of any errors the program exits and an error message with cause of the error is displayed on the screen.

Utility functions:

convert (char destip[], unsigned short destport) : Converts the ip and port in parameters to the long mytpe field. Return value: long
extractport (unsigned long mtype): Returns port in NBO form. It is extracted from mtype that was passed as parameter.
extractip (unsigned long mtype): Returns ip in NBO form. It is extracted from mtype that was passed as parameter.

Running the program:

```
./local_server <ip of the machine>
```

```
./driver(on another terminal)
```