This document is to define the protocol to be used between the client and server processes for the Sup chat application. It is divided into sections based on a standard flow of action. It includes the proper formatting of supported messages and potential error code responses.

General

All messages are sent in ASCII text. Variable (e.g. actual chat messages) or unprintable ASCII (e.g. EOT) are enclosed in brackets, <>, to denote appropriate substitutions. All messages are terminated with an end of transmission character, denoted “<EOT>”, ASCII value 0x04.

The server shall respond to all client messages with a status message. These status messages dictate the success or failure of the last sent message of the client. The universal (universal meaning it is not dependent upon the state of the application) success status message is:

```
status<space>000<space>OK<EOT>
```

Universal failure message include:

```
status<space>100<space>Malformed Message<EOT>
status<space>200<space>Internal Error<EOT>
```

In general, 100 level error message indicate a client problem, 200 level error message indicate an internal error on the server.

Login

After successful establishment of a TCP connection, client shall send a login message to the server of the form:

```
login<space><username><EOT>
```

If the login is successful, the server shall respond with the universal success message.

There are possible errors on the login, and the server should respond with as specific of error as possible. Some of the possible errors include:

```
status<space>101<space>Username Taken<EOT>
status<space>102<space>Username Invalid<EOT>
```

Contact Query

Clients can query a list of all online contacts. The message format is:

```
get<space>contacts<EOT>
```

The server shall respond with a status message, and if successful, a subsequent message including a list of space-separated contact names. The format for the contact list message is:

```
contacts<space><username1><space><username2><...><EOT>
```

The client can also query the status of a particular contact. The message to do so has the form:

```
contact<space><username><EOT>
```

The server shall respond with the following message if the contact is online:

```
status<space>001<space>Online<EOT>
```

If the queried contact is not online, the response shall instead be:

```
status<space>103<space>Offline<EOT>
```

Send and Receive Chat Message

Once logged on, a client can send chat messages to other clients. These messages are routed to the destination user by the server application. A chat message from one client to another shall have the format:

```
send<space><to username><space><message><EOT>
```

The server will then forward the message to the destination contact, and the forwarded message shall have the format:

```
recv<space><from username><space><message><EOT>
```

The server shall respond to the initiating client with the universal success message if it was able to deliver the message. Here are some specific error cases:

```
status<space>104<space>Messaged User Not Online<EOT>
```

Log Off

When the user wishes to exit the client application, the client shall notify the server of the log off. This message shall have the format:

```
quit<space><username><EOT>
```

The user should also wait to receive the status message from the server before ending the process.