Computer Networks (*Redes de Computadores - RCOMP*) - 2023/2024

Integrative Project of the 4th Semester

Application Protocol

The application protocol described in this document is of mandatory use for all the communications between the client applications (Candidate App and Costumer App) and the Follow Up Server. Direct interaction between the client applications (Candidate App and Costumer App) and the database server is not allowed.

1. Protocol description

- It´s a TCP (Transmission Control Protocol) based client-server protocol.
- The client application takes the initiative of establishing a TCP connection with the server application, for such the client application is required to know (IP address or DNS name) the node where the server application is running and the TCP port number where the server application is accepting TCP connections.
- After the TCP connection is established, the connected applications exchange messages with the format described in Section 2.
- Once established, the TCP connection between the client application and the server application is kept alive and is used for all required data exchanges (requests and responses) while the client application is running.
- All message exchanges between the client application and the server application must follows a very restrict client-server pattern: the client application sends one request message, and the server application sends back one response message.

2. Messages format

Every data exchange through the TCP connection (requests or responses) must comply with the bytes sequence description in Table 1, this is the message format version one. This message format is not expected to change during this project development.

Field	Offset (bytes)	Length (bytes)	Description
VERSION	0	1	Message format version. This field is a single byte and should be interpreted as an unsigned integer (value 0 to 255). The present message format version number is one.
CODE	1	1	This field identifies the type of request or response, it should be interpreted as an unsigned integer (value 0 to 255). See Section 3.
DATA1_LEN_L	2	1	These two fields are used to specify the length in bytes of the following DATA1 field. Both these fields are to be interpreted as unsigned integer numbers (value 0 to 255). The length of the DATA1 field is to be calculated as: `DATA1_LEN_L + 256 x DATA1_LEN_M` If the resulting value is zero, it means DATA1 does not exist, and the message has ended at this point.
DATA1_LEN_M	3	1	
DATA1	4	-	First chunk of data, contains data to meet the specific needs of the participating applications, its existence and the content depend on the message's code (type of request or response).
DATA2_LEN_L	-	1	These two fields are used to specify the length in bytes of the following DATA2 field. Both these fields are to be interpreted as unsigned integer numbers (value 0 to 255). The length of the DATA2 field is to be calculated as: `DATA2_LEN_L + 256 x DATA2_LEN_M` If the resulting value is zero, it means DATA2 does not exist, and the message has ended at this point.
DATA2_LEN_M	-	1	
DATA2	-	-	Second chunk of data, contains data to meet the specific needs of the participating applications, its existence and the content depend on the message's code (type of request or response).
…			

Table 1 - Message format

In this message format, the transport of any number of chunks of data (DATA1, DATA2, DATA3, …) is supported, all messages end with two bytes with value zero that state that the next chunk of date does not exist, and the message has ended.

Every message terminates with a sequence of two zero bytes. A message may not carry any data, such a message is only four bytes long.

3. Message codes

Table 2 presents the set of initial message codes to be supported. As the project is developed, new message codes are supposed to be added by the team to meet the needs of new features.

After establishing the TCP connection with the server application, the client application must undertake a user authentication procedure by sending an AUTH request carrying a username and a password. Until there's a successful user authentication, the server application will refuse any other request other than AUTH, COMMTEST or DISCONN.

CODE	Type	Meaning
0	**Request**	**COMMTEST** – Communications test **request** with no other effect on the server application than the response with an ACK message. This request has no data.
1	**Request**	**DISCONN** – End of session **request**. The server application is supposed to respond with an ACK message, afterwards both client and server applications are expected to close the session (TCP connection). This request has no data.
2	**Response**	**ACK** – Generic acknowledgment and success **response** message. Used in response to a successful request. This response contains no data.
3	**Response**	**ERR** – Error **response** message. Used in response to unsuccessful requests that caused an error. This response message may carry a human readable phrase explaining the error. If used, the phrase is carried in the DATA1 field as string of ASICII codes, it's not required to be null terminated.
4	**Request**	**AUTH** – User authentication **request** carrying the username in DATA1 and the user's password in DATA2, both are strings of ASICII codes and are not required to be null terminated. If the authentication is successful, the server application response is ACK, otherwise it's ERR.
…		

Table 2 - Message codes