

Prof. Dr. Marcus Schöller

Subtask 1.1: Code analysis (5 points)

Analyze the provided source code in case the user issues a legal command. So far, the actual command communication is not implemented (you have to do this in subtask 1.2). Only the preparatory step to request a cookie from the server is implemented.

You can use a debugger to observe the code during operation. You have to document protocol stack initialization and the cookie request method all the way until a cookie is available to the client or the request got rejected. Document your findings and visualize them in a sequence diagram. Use **class names** as actors.

Subtask 1.2: Command Protocol: print command (5 points)

Once the client has a valid cookie, it can send command messages to the server. Implement all functionality of the client to send a **print command** message to the server and to receive the respond from the server.

1. Completion of send method in CProtocol.
 - a. The client must only send command messages if a valid cookie was received.
 - b. Create a status command message object (see below).
 - c. Send to server. The PHY config is stored as an attribute in CProtocol.
2. Implementation of receive method in CProtocol.
 - a. For any command message sent, the client waits at most three seconds for a response from the server. Call the appropriate receive method of the PHY layer.
 - b. Call the message parser to create a CP message object from the String object received according to the protocol specification. If the parser throws an exception, the message shall be discarded.
Note: you can use the receive method of the cookie response message as a template.
 - c. Check that the response matches the command message by comparing the message id of the received message with id of the sent message.
3. Create and implement a message classes to create and parse command messages.
 - a. To calculate the crc-checksum use the CRC32 class of Java (<https://docs.oracle.com/javase/9/docs/api/java/util/zip/CRC32.html>)
 - b. Add getter/setter methods as needed by other tasks.

Advanced feature for subtask 2

In case you finish early on this part and you want to improve your implementation further you might want to enhance your implementation in the following way:

1. The message parsing should be implemented with regular expressions (Regex). See the Java Regex tutorial for details:
<https://docs.oracle.com/javase/tutorial/essential/regex/intro.html>

This advanced feature is not mandatory to implement and will not provide any points for the hands-on. It is rather experience you will gain from this 😊.