Connection Manager Assignment

Deliver to: noha_gamal@mentor.com

Overview

Your team is working on a new tool, a part of this tool is supposed to manage the connection between a server and a number of clients.

Details

- Only one server first comes up, then a number of clients can come up and connect to the server.
- The server constructs a message with certain format and broadcasts this message to all connected clients.
- The message contains a prefix which could be either a four character prefix "abcd" or an eight character one "abcdefgh".
- Then a custom body that could be of length 1 up to 30 characters will be appended after the prefix.
- User chooses which prefix to use and passes the message body. Then the user can optionally choose to add a suffix or not.
- If a suffix to be added, the user should select one of two forms, either a five character suffix "ABCDE" or a ten character one "ABCDEFGHIJ".
- The difference is that the suffix is not appended to the message. Instead, it should overwrite the last 5/10 characters of the message body.
- As a first increment, one developer wrote the piece of TCL code below implementing a server and only two clients.
- S/He passes the code to you along with an example code s/he used to test it.
- You are required to review the below code and give comments on its structure, maintainability, any potential bugs…

```
namespace eval ConectionManager {
    variable ServerUp 0
    variable client1Up 0
    variable client2Up 0
    variable clientlist ""
    variable Msg ""
    variable prefix1 "abcd"
    variable prefix2 "abcdefgh"
    variable suffix1 "ABCDE"
    variable suffix2 "ABCDEFGHIJ"

    proc StartServer {} {set ConectionManager::ServerUp 1}
    proc StartClient1 {} {set ConectionManager::client1Up 1}
    proc StartClient2 {} {set ConectionManager::client2Up 1}
    proc ConnectClient1 {} {lappend ConectionManager::clientlist 1}
    proc ConnectClient2 {} {lappend ConectionManager::clientlist 2}
    proc CloseServer {} {set ConectionManager::ServerUp 0}
    proc CloseClient1 {} {set ConectionManager::client1Up 0}
    proc CloseClient2 {} {set ConectionManager::client2Up 0}

    proc ConstructMsg {prefixType msgBody suffixType} {
           set msg ""
           # Add prefix
           switch $prefixType {
```

```
                1 {
                        set msg $ConectionManager::prefix1
                        set maxMsgLength 34
                }
                2 {
                        set msg $ConectionManager::prefix2
                        set maxMsgLength 38
                }
                default {puts "Unknow prefix !"}
                }

                # Add message body
                set msg "$msg$msgBody"
                # Trim message
                if {[string length $msg] > $maxMsgLength} {
                        set msg [string range $msg 0 $maxMsgLength]
                }
                set msglength [string length $msg]

                # Add suffix
                switch $suffixType {
                        0 {set ConectionManager::Msg $msg}
                        1 {set ConectionManager::Msg [string replace $msg [expr $msglength
                        - 5] $msglength $ConectionManager::suffix1]}
                        2 {set ConectionManager::Msg [string replace $msg [expr $msglength
                        - 10] $msglength $ConectionManager::suffix2]}
                        default {puts "Unknow suffix !"}
                }
        }

        proc BroadCast {} {
                foreach client $ConectionManager::clientlist {
                        # Some code to send message
                        puts "message $ConectionManager::Msg was sent to client $client"
                }
        }
}
#_____________________________________________________________________________
# Example for testing
ConectionManager::StartServer
ConectionManager::StartClient1
ConectionManager::ConnectClient1
ConectionManager::ConstructMsg 1 "123456789" 0
ConectionManager::BroadCast
ConectionManager::StartClient2
ConectionManager::ConnectClient2
ConectionManager::ConstructMsg 2 "12345678901234567890123456789O" 1
```

Expected Deliverables

1. Detailed code review.
2. Bug reports for all your findings.

Evaluation Criteria

1. Bug reports' structure.
2. The coverage of your code review.

Guidelines

Deadline: 2 days.