GL C/C++ BaseCamp 2022 Task 1

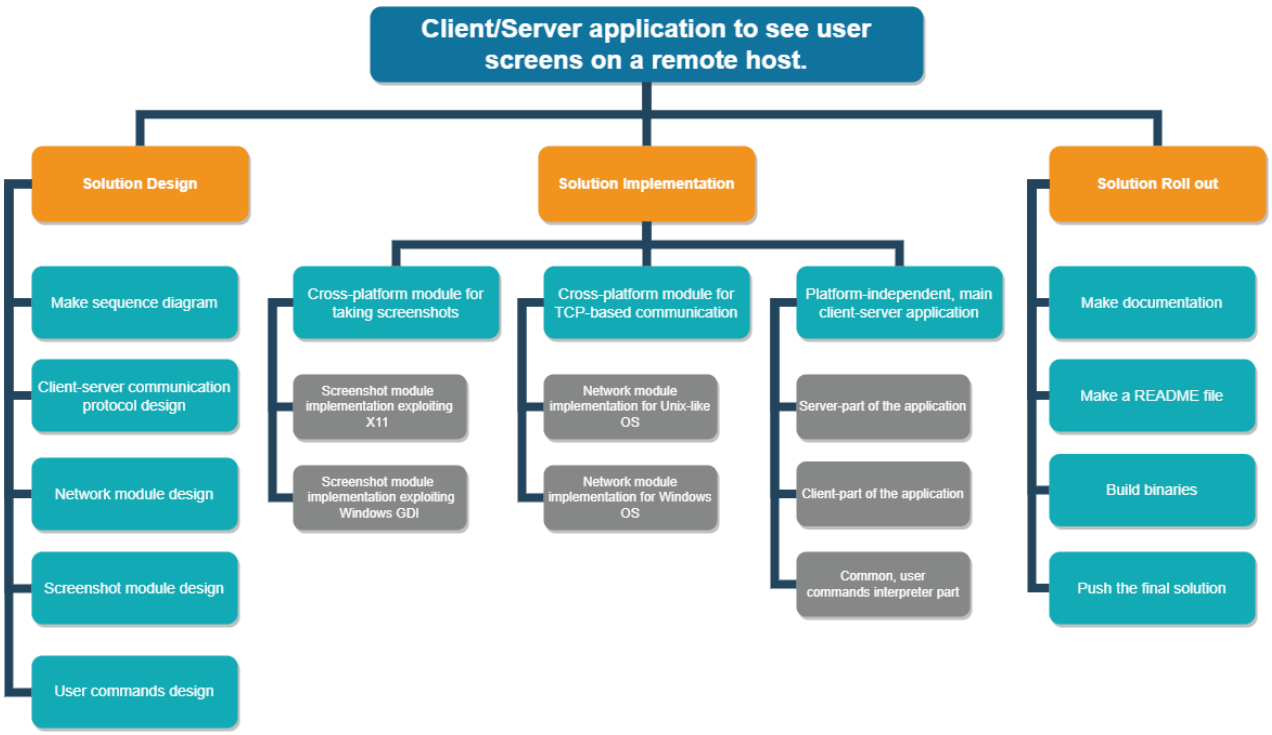
Myroslav Mishchuk

Task

Client/Server application to see user screens on remote host.

- Client has to make "print-screen" each 60 seconds.
- All screens must be stored in client app folder.
- Each screen name consists of date, when it has been taken + number of screen.
  Ex: 21.07.2018_15.png 21.07.2018_16.png ...etc
- Server can request screens by those names.
- File must be transferred to server via 5555 port.

WBS



Comments:

Cross-platform module for taking screenshots: Implementation for Unix-like systems will exploit X11 (X Windows Systems, Xlib). X11 was chosen because almost all Linux desktop systems are built on top of it. For Windows-based implementation, Windows GDI will be used.

Cross-platform module for TCP-based communication: a high-level framework that works over POSIX/WinSock TCP-sockets. It implements the designed SFRC (simple remote function call) application-level protocol (discussed later).

User commands design: design of how the user will run the application from the command line.

Estimations:

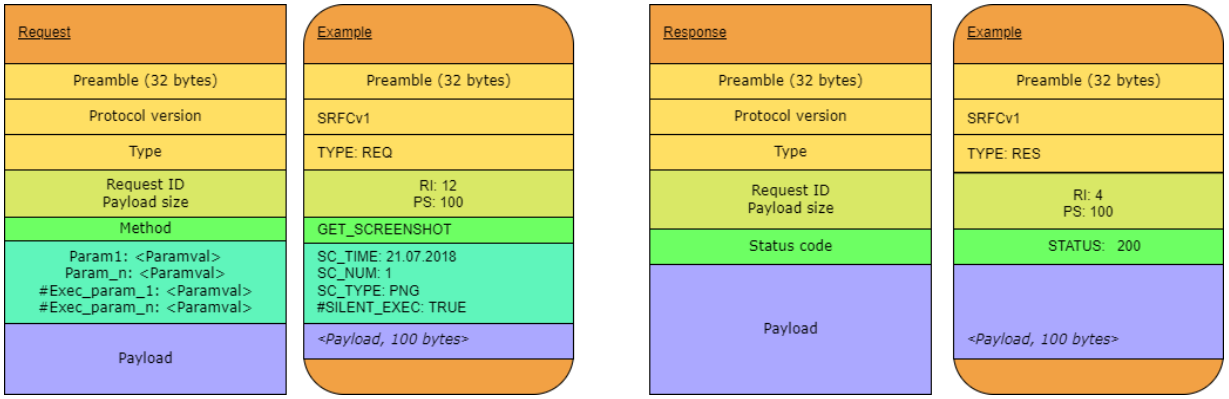
Solution Design: 2 days
Solution Implementation:
- Cross-platform module for taking screenshots: 1 day
- Cross-platform module for TCP-based communication: 3-4 days
- Platform-independent, main client-server application: 2 days

Solution Roll out: 1 day
Total: 9-10 days

Client-Server communication protocol

The SRFC (simple remote function call) protocol was designed as an application-level protocol for client-server communication, built over TCP. This protocol was designed for bi-directional communication with asynchronous remote function invocation, with the subsequent response with the function return value.

Header structure:



Sequence UML for Client-Server communication