Software design document (SDD) of P2P Chat System project

ASMA SOUFI / YUANBO WANG
4IR – Group A2

Table of contents

1. Overview of requirements & design decisions

The purpose of this software design document (SDD) aims to describe the architecture as well as design pattern decisions that we have made in order to build the Peer-to-Peer Chat System.

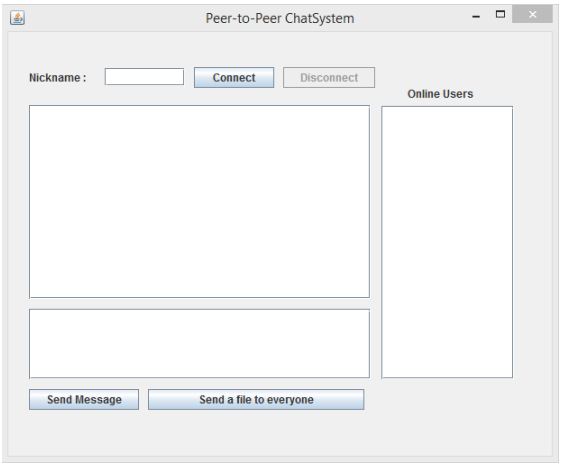
The document is devided into four parts. In the first place, high level decomposition diagrams will show possible interactions between three main components of the chat system. Secondly, detailed sequence diagrams are used to specify the collaboration of internal components inside chatNI and chatGUI. Thirdly, the component diagram will show each component (e.g. subsystem) inside the system, and finally a white box class diagram shows the whole map of classes that are to be implemented.

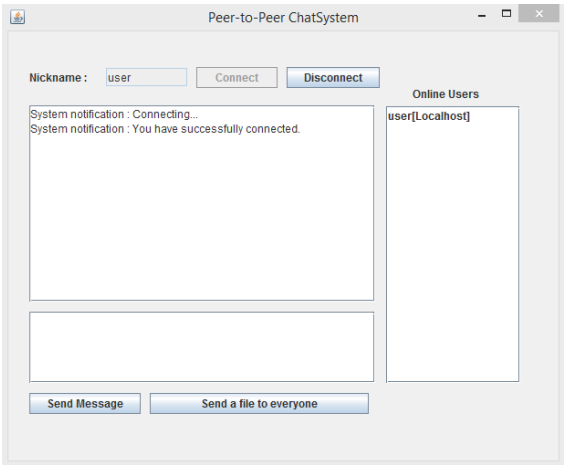
Requirements

Our P2P chat system should allow every user to connect to the chat system with a valid nickname. When a user connects to the system, the list of the other connected users is presented. This list includes connected user names and their IP address. Only connected users are able to use different functionalities of the chat system functions, and when any user connects or disconnects, other users have to be informed about it. A user can communicate with another user (or all other users) by sending messages or files, he has to select the remote user from the connected users' list.

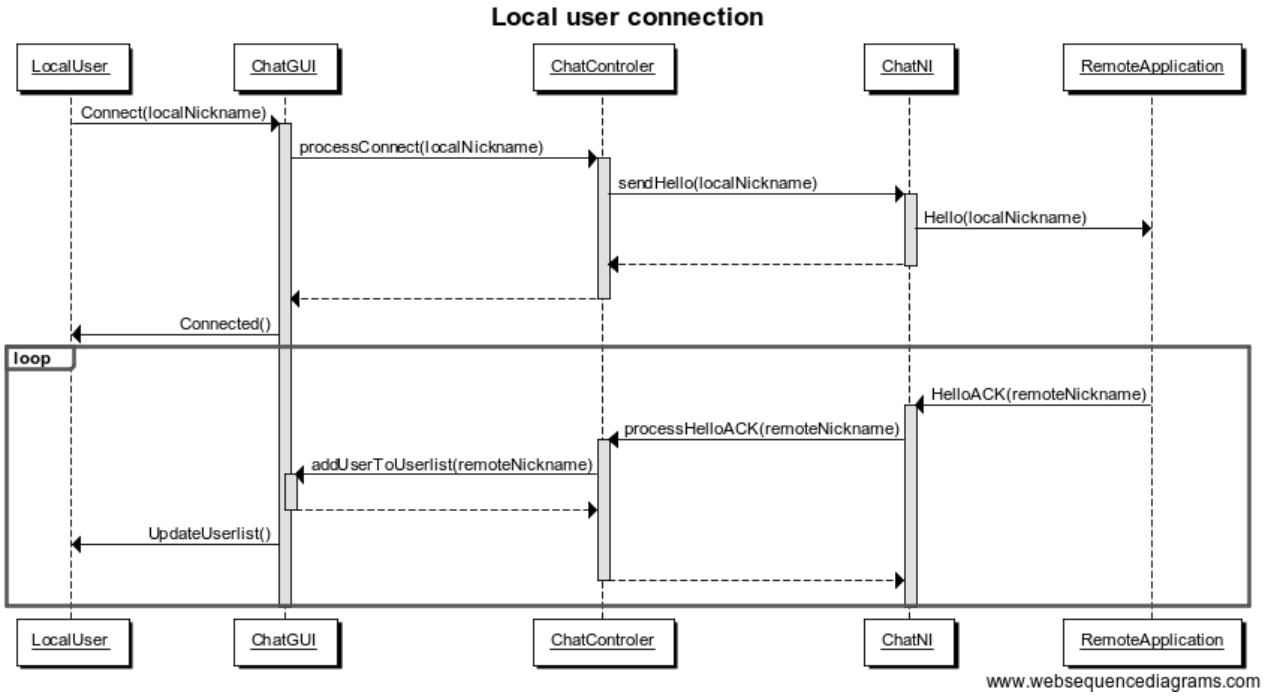
Decisions

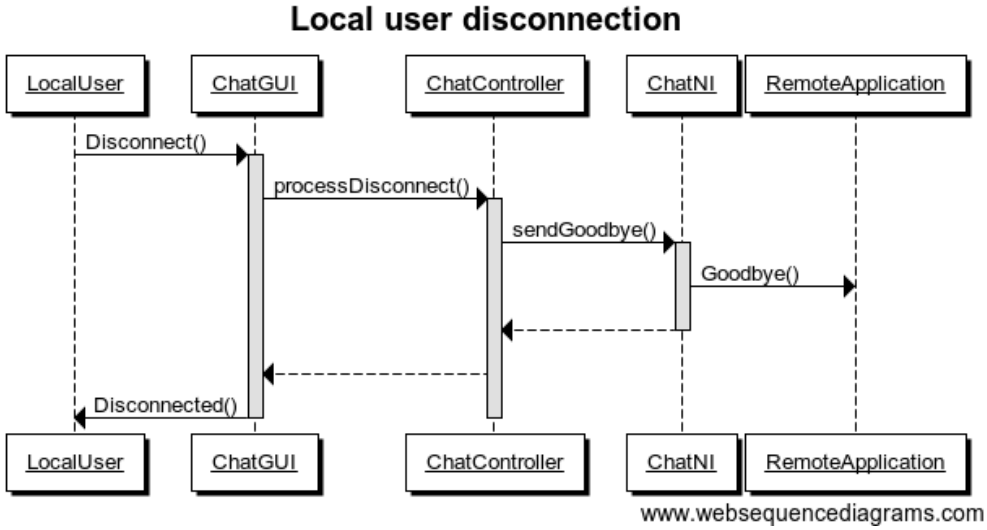
We decide to implement our chat system using two main design patterns : MVC design pattern and Observer design pattern. For MVC design pattern, we define a chatController as “Controller” role, chatGUI and chatNI as “View” role, and chatModel as “Model”. For the observer design pattern, we decide to make chatNI and chatGUI as observers, and FileReceiver, UDPreceiver as well as chatModel as observables. We alse decide to use FileChooser to facilite the file sending. The UI of our chat system is as follows :



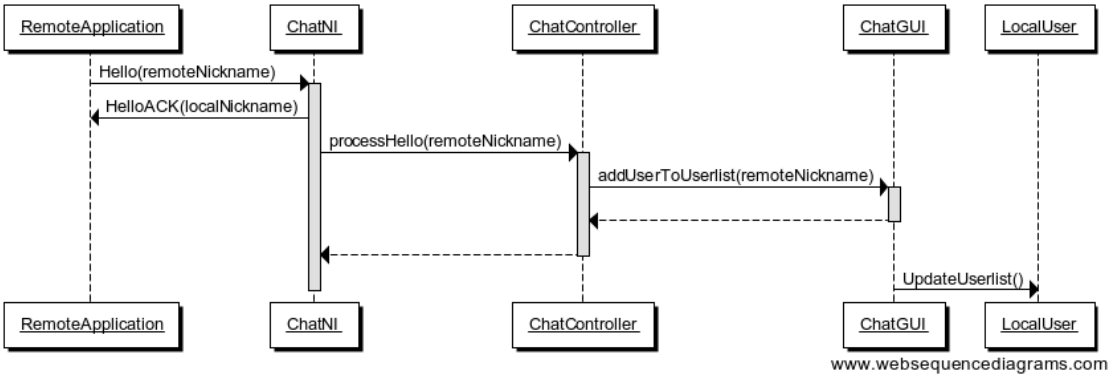


2. High level decomposition diagram

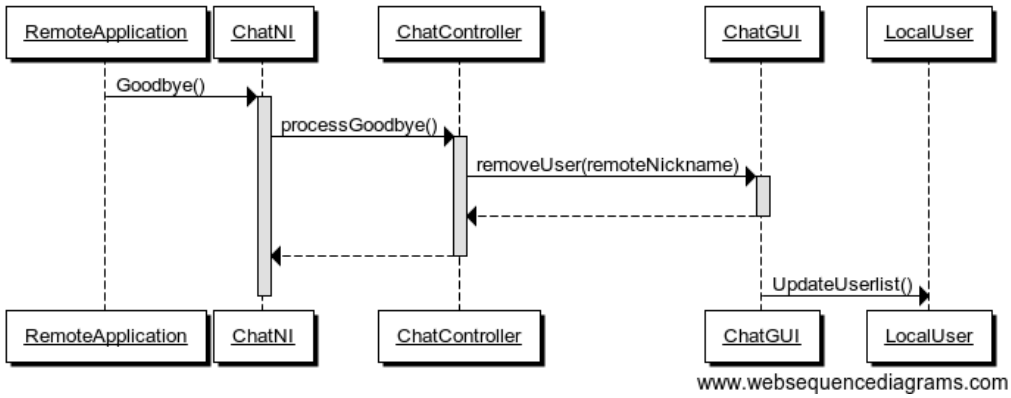




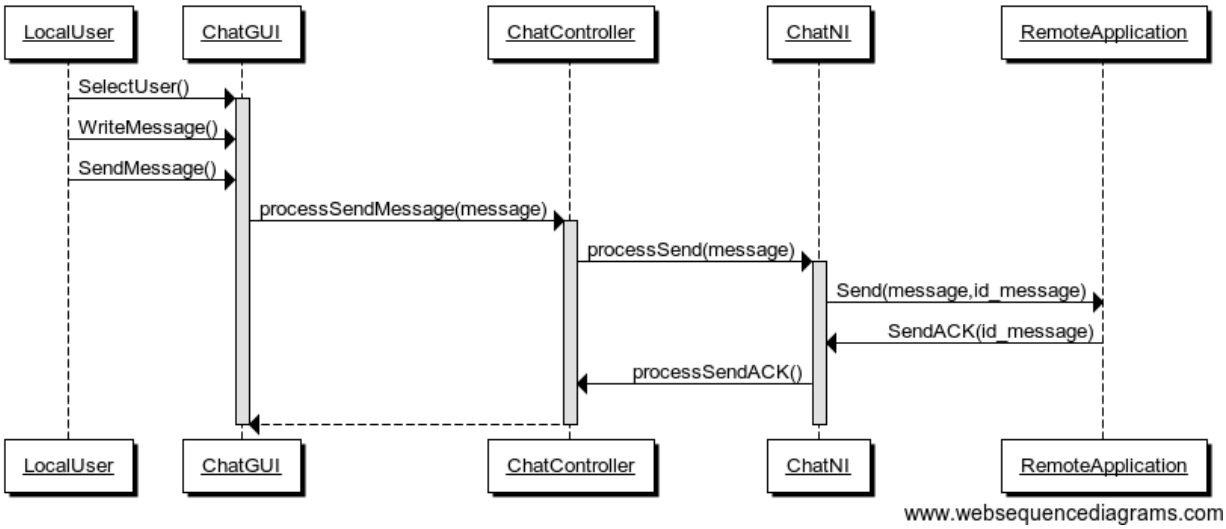
Remote user connection



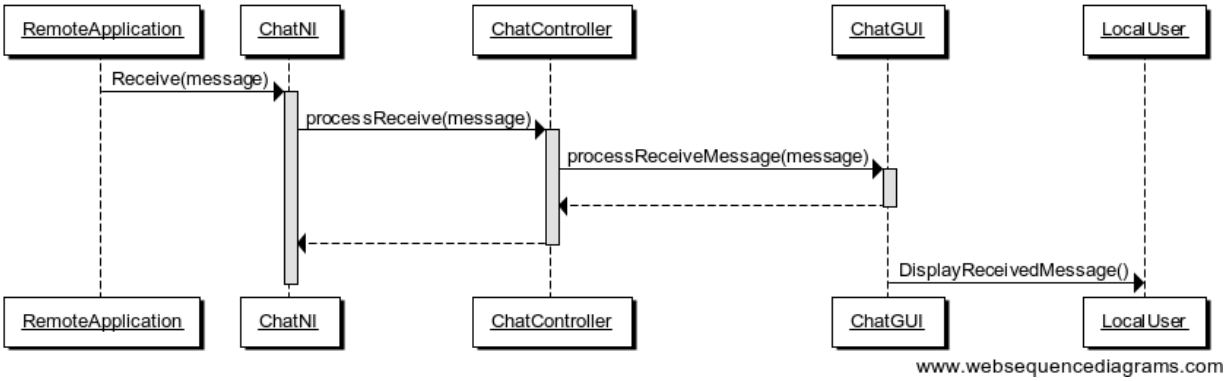
Remote user disconnection



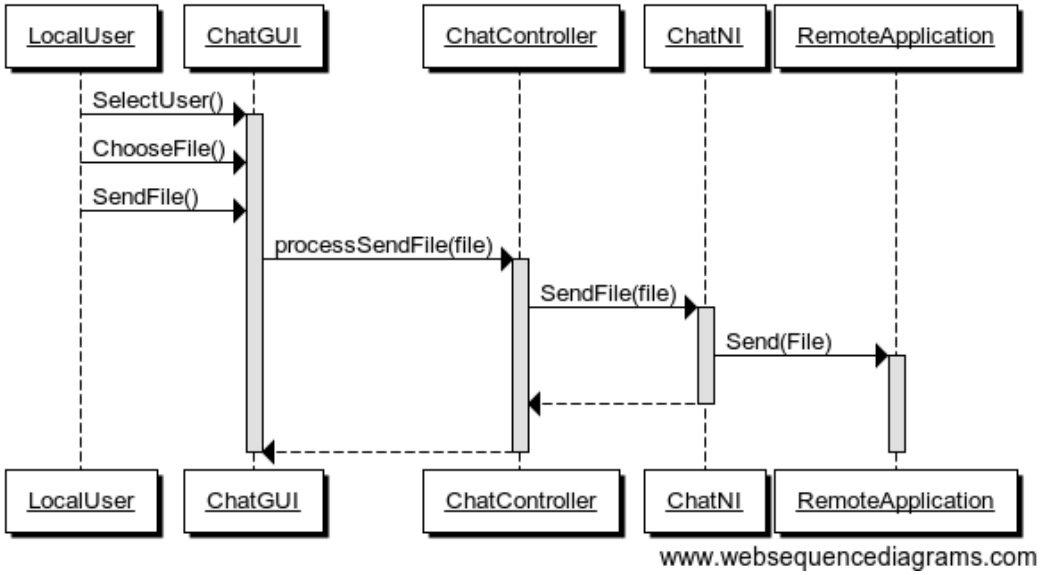
Local user sends messages



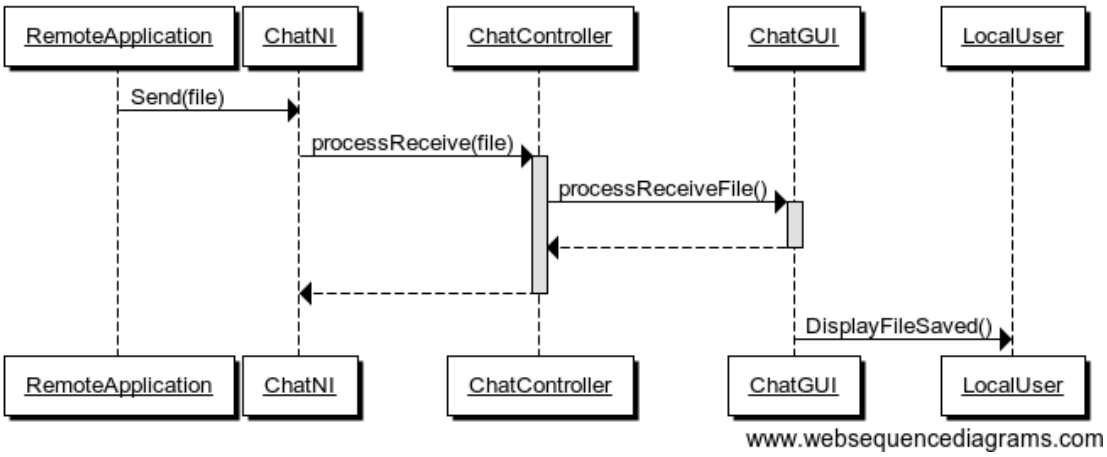
Local user receives messages



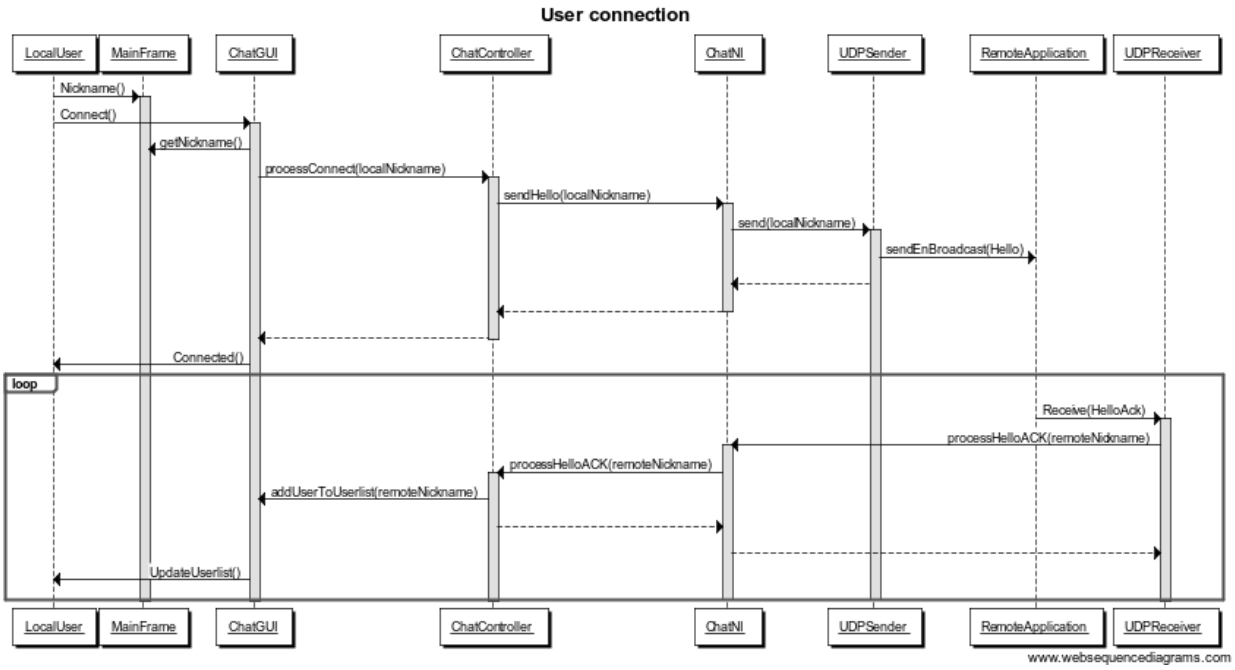
Local user sends files



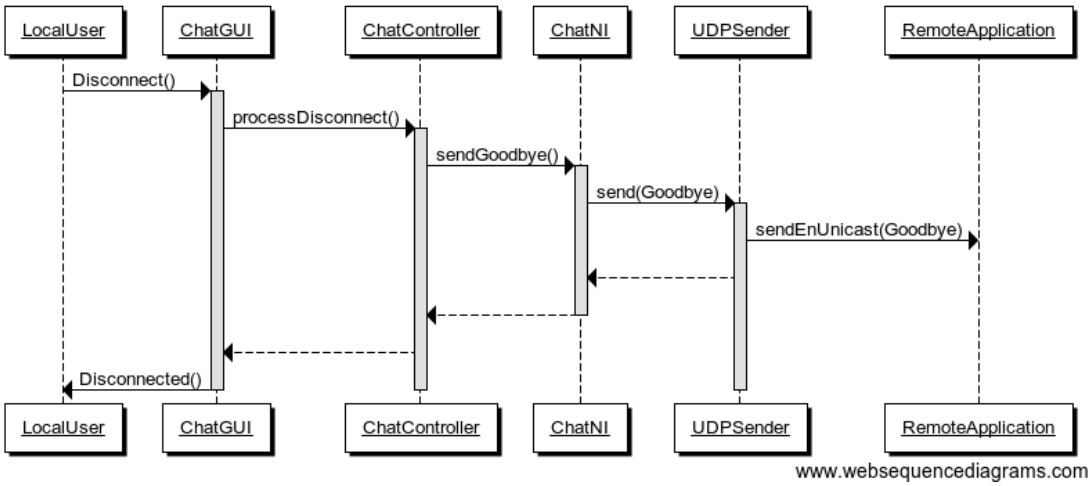
Local user receives files



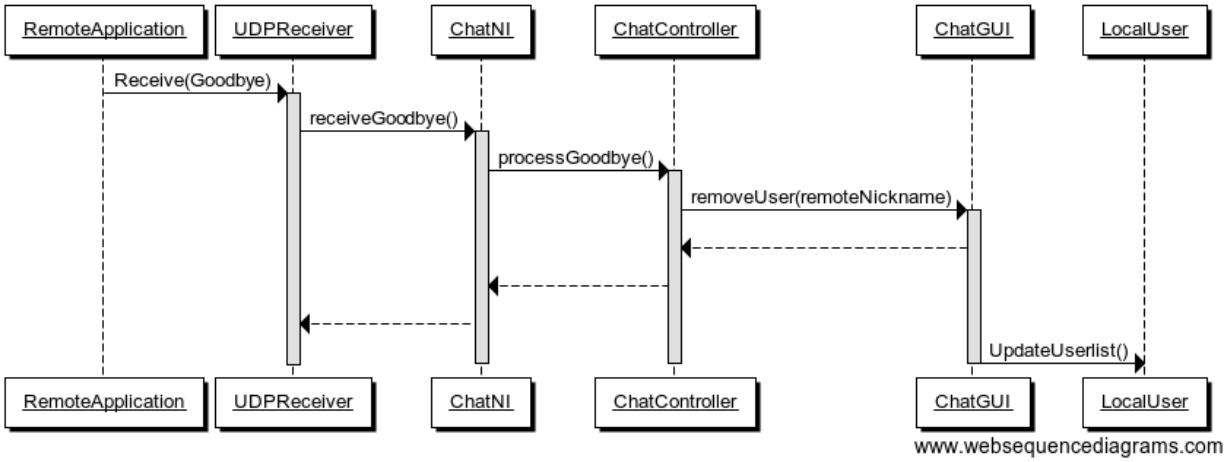
3. Sequence diagram (whitebox)



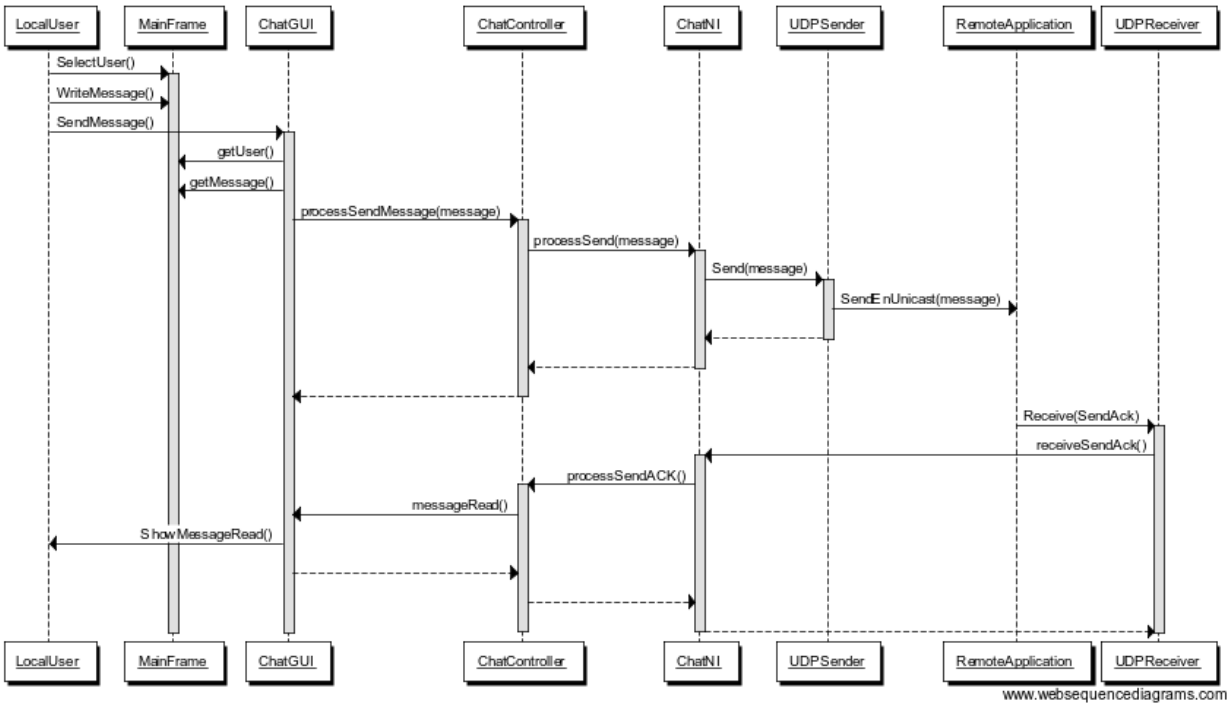
Local user disconnection



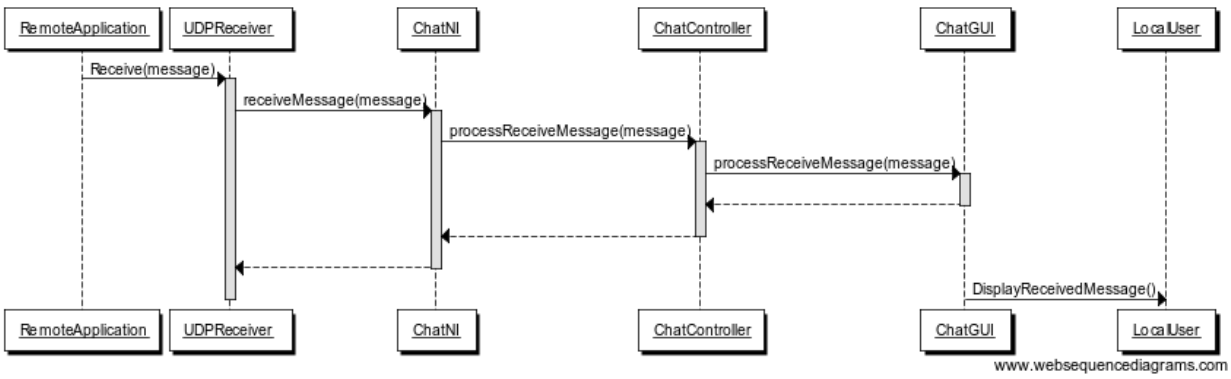
Remote user disconnection



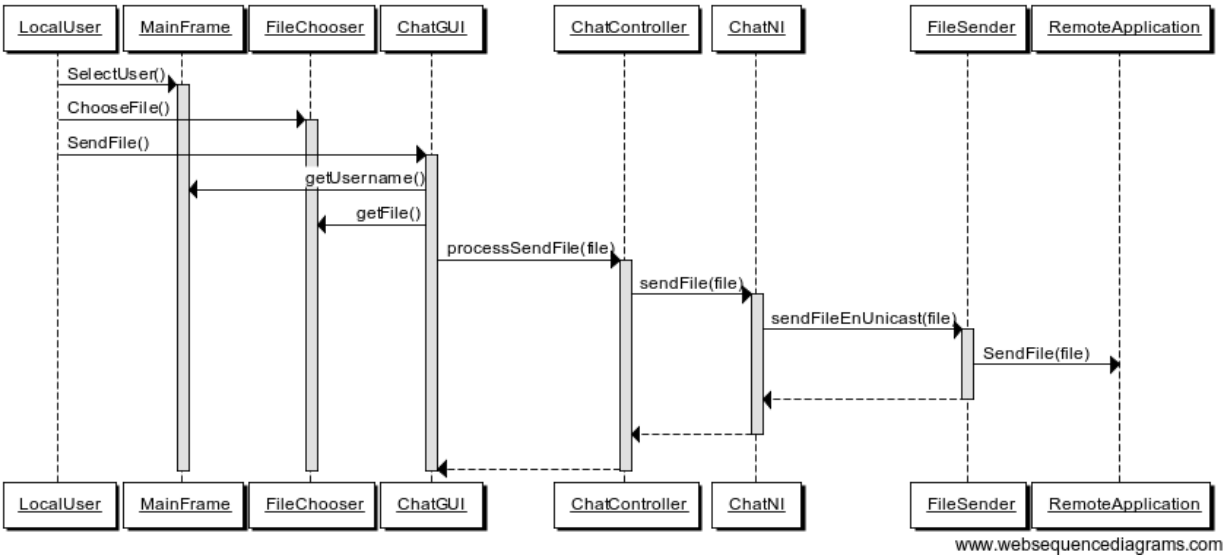
Local user sends messages



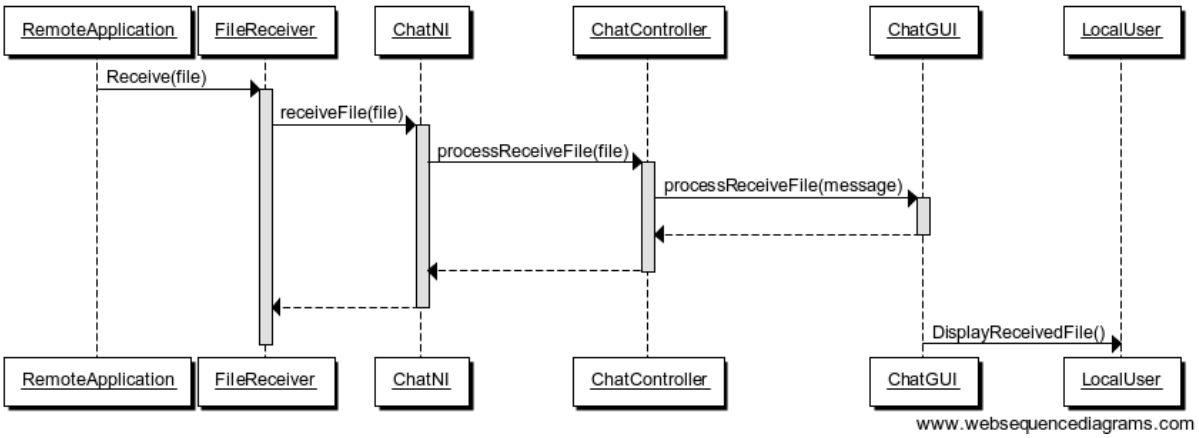
Local user receives messages



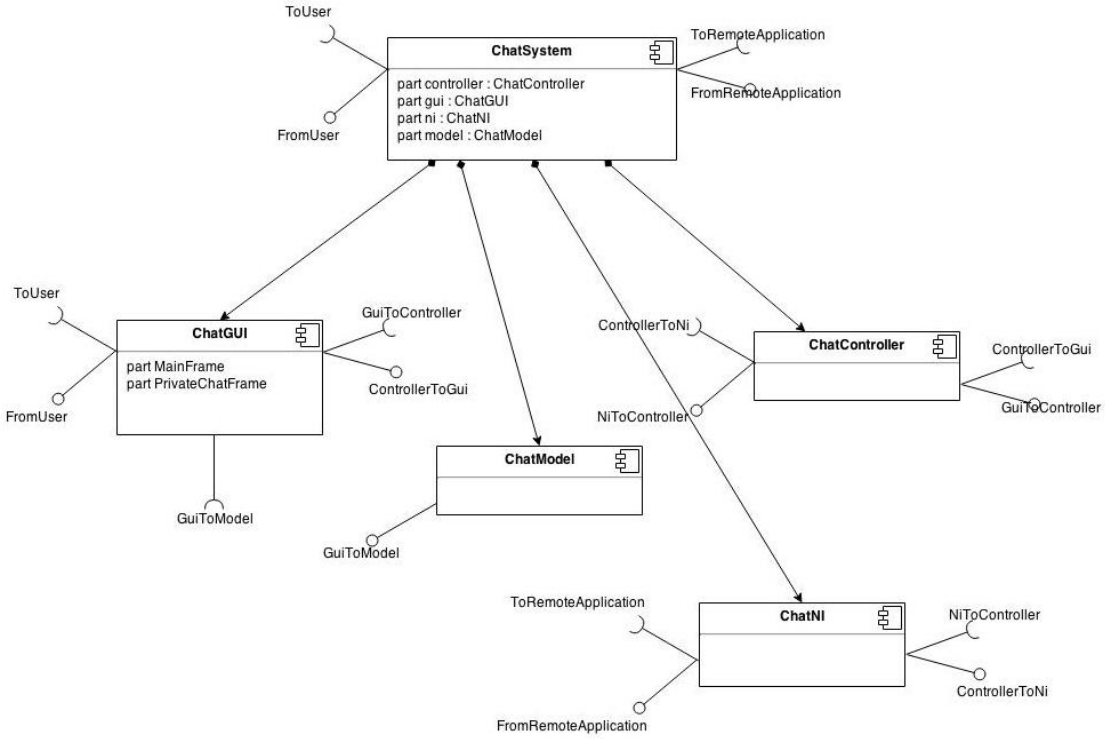
Local user sends files



Local user receives files



4. Component diagram (whitebox)



5. Class diagram (whitebox)