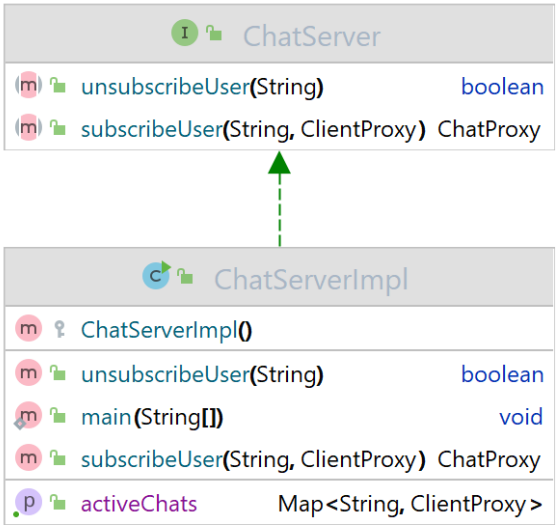
Verteilte Systeme im Sommersemester 2022

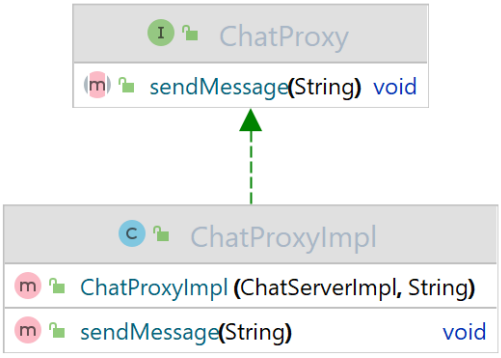
Patrick Felschen, Matr. Nr. 932056
Julian Voß, Matr. Nr. 934505

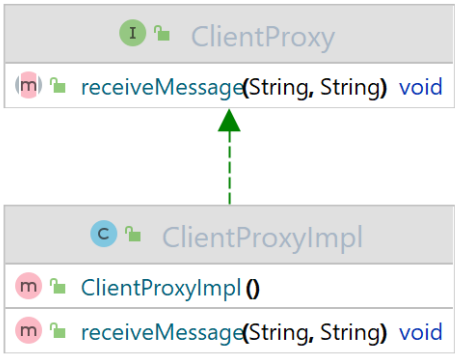
Osnabrück, 18.05.2022

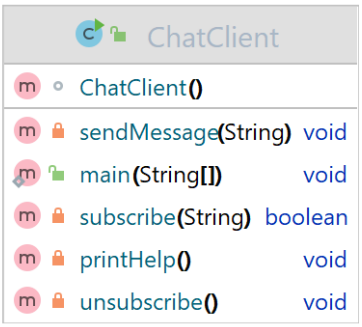
Aufgabenblatt 6

Aufgabe 1 – Implementierung









Anmelden des Servers in der Registry

Beim Starten des Servers wird die Registry unter Angabe des Standardports (1099) gestartet. Es wird ein neues Serverobjekt erstellt, welches der Registry unter dem Namen „ChatServer“ hinzugefügt wird.

```
public static void main(String[] args) throws RemoteException {
    Registry registry = LocateRegistry.createRegistry(Registry.REGISTRY_PORT);
    ChatServerImpl chatServer = new ChatServerImpl();

    registry.rebind("ChatServer", chatServer);

    System.out.println("Server started");
}
```

Anmelden des Clients in der Registry

Der Client sucht sich die im Server angelegte Registry unter dem Standardport. Danach kann der Server mit dem vorher festgelegten Namen in der Registry gefunden werden.

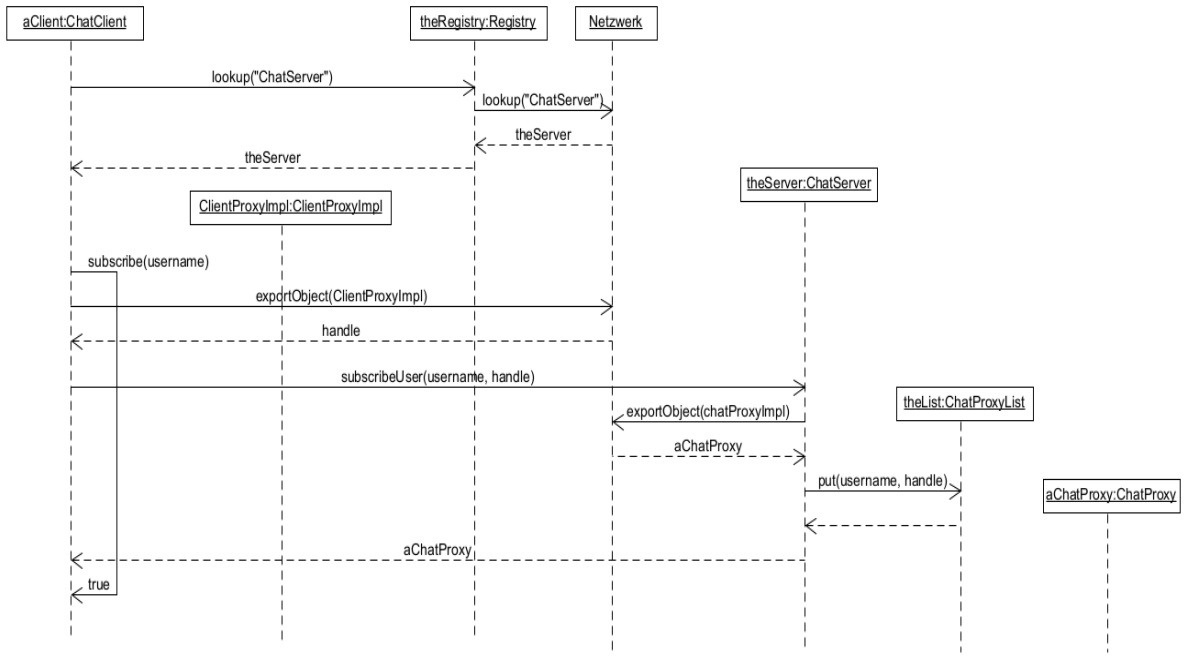
```
this.registry = LocateRegistry.getRegistry(Registry.REGISTRY_PORT);
this.chatServer = (ChatServer) registry.lookup("ChatServer");
```

Registrieren des Clients im Server

Auf der Client Seite wird ein Handle vom Typ ClientProxy angelegt, welcher im Server genutzt wird, um Verbindungen den Nutzernamen zuordnen zu können. Danach wird die Serverfunktion *subscribeUser* mit dem Nutzernamen und Handle als Parameter aufgerufen. Der Server legt eine neues ChatProxy Element an, welches die Kommunikationsschnittstelle für die Nachrichtenverteilung darstellt. Das ChatProxy Element wird an den Client zurückgegeben, worüber dieser Nachrichten an andere Nutzer senden kann. Der Client-Handle und der Nutzername werden im Server in einer Map abgespeichert.

```
// Client
ClientProxyImpl clientProxy = new ClientProxyImpl();
try {
    ClientProxy handle = (ClientProxy)
UnicastRemoteObject.exportObject(clientProxy, 0);
    this.chatProxy = chatServer.subscribeUser(input, handle);
} catch (RemoteException e) {
    throw new RuntimeException(e);
}
// Server
ChatProxyImpl chatProxyImpl = new ChatProxyImpl(this, username);
ChatProxy chatProxy = (ChatProxy)
UnicastRemoteObject.exportObject(chatProxyImpl, 0);

chatProxyList.put(username, handle);
```



Senden von Nachrichten

Nachdem der ChatClient sich beim Server angemeldet hat, kann über die „sendMessage“-Funktion des ChatProxys eine Nachricht veröffentlicht werden.

In der ChatProxyImpl werden sich zunächst alle aktiven (registrierten) ClientProxys vom ChatServer geholt. Danach wird die „receiveMessage“-Funktion mit dem Absender und der Nachricht als Parameter auf jedem ClientProxy ausgeführt.

```
// ChatClient
chatProxy.sendMessage(message);

// ChatProxyImpl
new HashMap<>(chatServer.getActiveChats()).forEach((username, clientProxy) -> {
    try {
        if(!this.username.equals(username)) {
            clientProxy.receiveMessage(this.username, message);
        }
    } catch (RemoteException e) {
        try {
            chatServer.unsubscribeUser(username);
        } catch (RemoteException ex) {
            throw new RuntimeException(ex);
        }
        System.out.println(this.username + " konnte nicht erreicht werden.");
    }
});
```