TW-Mailer Extended

BIF3 – WS2021

Florian David Lederer | if20b116

Johannes Maximilian Rathgeb | if20b145

Client and Server Architektur

Client:

Der Client verbindet sich mit einer IP-Adresse und einem Port mit dem Server. Danach können folgende Befehle eingegeben werden:

* SEND: Der User gibt den Empfänger, Betreff und den Inhalt der E-Mail ein. Diese wird anschließend an den Server geschickt. Der Sender wird automatisch anhand der eingeloggten Session ID in der Mail ergänzt.
* LIST: Der User gibt einen existierenden Benutzer an. Anschließend werden alle Betreffe mit Nummerierung aufgelistet.
* READ: Der User gibt einen existierenden Benutzer und eine Message-ID an. Diese wird anschließend vollständig ausgegeben
* DEL: Der User gibt einen existierenden Benutzer und eine Message-ID an. Diese wird vom Server gelöscht.
* QUIT: das Programm wird beendet.
* LOGIN: Der User gibt seine UID ifxxbxxx ein und das dazugehörige Passwort. Ist der Login erfolgreich, wird der User eingeloggt und ihm stehen sämtliche obenstehende Commands zur Verfügung. Andernfalls kann er nur <LOGIN> oder <QUIT> ausführen.

Server:

Der Server wird gestartet, indem ihm ein Port mitgegeben wird. Stimmt dieser mit dem Client überein, so stellen beide miteinander eine Verbindung her. Zusätzlich wird ein Mailverzeichnis mitgegeben, in das die User und die zugehörigen Mails gespeichert werden. Für jeden User wird eine Index-Datei angelegt, welche die Nummerierung der E-Mails mitzählt. Dadurch kann jeder E-Mail eine eindeutige ID zugeordnet werden.

Funktion:

Der Client wartet auf einen Befehl. Anschließend werden die dazugehörigen Parameter eingegeben. Sollten diese nicht valide sein, muss der User seinen Input wiederholen. Sind alle Eingaben in Ordnung, werden diese gemeinsam an den Server geschickt. Sollte der Server alle Eingaben korrekt verarbeiten können, wird ein „Ok“ an den Client zurückgegeben. Wenn ein Fehler auftritt, bekommt der Client ein „Err“ zurück. Danach kann der User einen neuen Command eingeben.

Technologien

Bei der Umsetzung des Programms haben wir mit Visual Studio Code und einer WSL und in einer Linux Distribution gearbeitet. Das Programm wurde in C++ geschrieben. Für die Kommunikation zwischen Server und Client wurden Sockets verwendet. Für den Login wurde LDAP verwendet. Für die Parallelisierung haben wir mit Kindprozessen 🡪 fork() gearbeitet.

Entwicklungsstrategie

Zu Beginn haben wir versucht das Musterbeispiel aus den Unterlagen in Visual Studio Code einzufügen und fehlerfrei auszuführen. Danach haben wir schrittweise die diversen Commands implementiert. Wir haben zuerst den QUIT Command implementiert und danach mit dem SEND Command begonnen. Als dieser fertig implementiert war, hatten wir nur noch wenig Probleme bei der Umsetzung der anderen Commands, da diese ähnlich funktionieren, wie der SEND Command. Bei der Erweiterung des Mailers haben wir den Programmdurchlauf zuerst mit fork() parallelisiert, danach die LDAP Verbindung hergestellt und somit den Login implementiert. Zum Schluss wurde der Send Command noch angepasst, damit der Sender automatisch der eingeloggte User ist.