TW mailer

Der TW-Mailer ist in 2 Files aufgeteilt twmailer-server.cpp und twmailer-client.cpp

Beide Files sind etwas schwer anfangs zu verstehen da sie sehr viel redundant code haben. Es wurde bei der Abgabe nicht auf Optimierung geachtet. In der End Abgabe wird dies dann der Fall sein.

Die Logik hinter den Funktionen ist jedoch einfach zu Erkennen. Der Server sowie der Client senden zuerst INFO Strings diese beinhalten Länge der Message, Anzahl der Packages, den Delimiter der benutzt wird um die Message zu parsen und den Typen der Message, also ob es SEND, READ, DEL etc… ist.

Beispiel für einen INFO string:

to\_string(tp.type) + "\n" + to\_string(tp.packageNUM) + "\n" + to\_string(tp.length) + "\n";

Der Delimiter wird hier automatisch erkannt durch die „\n“. Dieses Beispiel erklärt auch gleich wie Daten gespeichert werden. Es wird alles als ein Struct gespeichert. Alle Eigenschaften der Message egal ob der Originale Text oder auch die Metadaten. Alles wird in einem Struct gespeichert. Der Grund dahinter ist einfach. Es erleichtert die Returns und es vereinfacht die Verarbeitung von Messages, beim Parsen speziell.

Aufbau des Structs:

struct TextPreset  
{  
int packageNUM;  
std::string delim;  
int length;  
int type;  
std::string argument;  
std::string sender;  
std::string subject;  
std::string text;   
std::string infoString;  
std::string username;  
int ID;  
};

Nicht alle Variablen existieren bzw. werden in beiden Files benutzt. Aber der Großteil wird beim Server und beim User benutzt. Beim Server wird nur noch der Struct in einem Vector gespeichert. Beim User wird nichts gespeichert. Die variable type ist ein Integer wird aber mit einem Enum belegt.

enum type  
{   
SEND = 1, READ = 2, LIST = 3, DEL = 4, QUIT = 5, COMMENT = 0   
};

COMMENT wird nicht wirklich genutzt und ist eher als Placeholder gedacht

Funktionen die mit „parse“ beginnen parsen die Information die danach geschrieben steht. „parseINFO()“ parsed die Info String die zuerst geschickt wird, um dem Server bzw. dem Client die Information für den nächsten String zu schicken. „parseREAD“ parsed den Read String und immer so weiter.

Funktionen die mit „send“ beginnen senden immer zur jeweiligen anderen Datei. Also „sendMESSstring()“ sendet den Message string, „sendINFOstring()“ sendet den Info String etc…

Funktionen die „calc“ am Anfang haben „berechnen“ Strings bzw. rechnen im Hintegrund für spezielle Fälle. Zb. „calcINFOstring()“ erkennt die Länge des zu sendenden Strings und berechnet die „packageNUM“, den Delimiter etc…

Funktionen die mit „recv“ beginnen bekommen das von der Anderen Datei geschickte. Funktionen die mit „initialize“ beginnen sind etwas komplizierter sie initiieren bzw. starten den Save Prozess einer Message. Also wenn eine Message gespeichert werden soll. Es gibt zwei Versionen momentan einmal für Messages mit 1 Package und einmal mit Messages mit mehreren Packages. Dies wird später auch in eines verlegt.

Ansonsten gibt es noch Funktionen die keinen Speziellen Anfang haben und die sind dann nur für den User Input.

Beim Client die „userINPUTfindOpt()“ Funktion macht wie der Name sagt. Sie findet die vom User eingegebene Option zb: SEND und öffnet die richtigen Funktionen. Nochmals es ist nicht schön gemacht aber für die Basic Abgabe reicht es.