Poshet

Sabina-Alina Prodan

Universitatea "Alexandru Ioan Cuza", Iași, România

Rezumat Poshet este un client pentru serviciul POP3 în care se poate vizualiza, organiza și trimite e-mail. În continuare sunt detaliate atât tehnologiile utilizate, structura aplicației, cât și diverse detalii de implementare.

Cuvinte cheie: Mail · POP3 · SMTP · MIME · OOP · wxWidgets

1 Introducere

Poshet este un client cu interfață grafică, scris în C++ cu wxWidgets, pentru serviciile de mail electronic POP3 (pentru obținerea de mailuri) și SMTP (pentru trimiterea de mailuri). Printre funcționalitățile acestuia se numără: vizualizarea și organizarea mailurilor primite, trimiterea de mailuri noi cu atașamente, reply sau forwarding la mailuri primite.

2 Tehnologii aplicate

Proiectul este scris în C++ utilizând OOP. Pentru compilarea sa sunt folosite atât fișiere `Makefile`, cât și CMake. Proiectul include un fișier `README.md` care detaliază procesul de compilare.

Comunicarea în rețea cu serverele POP3 și SMTP se realizează prin intermediul funcțiilor POSIX. Protocolul ales la nivel de transport este TCP, fiindcă ambele protocoale la nivel de aplicație sunt bazate pe acesta.

Comunicarea cu baza de date SQLite creată local se realizează cu ajutorul librăriei SQLite3. În comparație cu alternative precum XML, o bază de date concretă permite interogări mai complexe și o organizare mai ușoară a mailurilor dintr-un inbox.

Pentru interfața grafică este utilizată librăria wxWidgets (în pofida altor librării similare precum Qt) datorită simplității sale, a structurii bazate pe OOP și în mod special a integrării sale facile cu Visual Studio Code și CMake.

Pentru analizarea și transformarea mailurilor din format MIME[3] într-un format afișabil în client, dar și pentru transpunerea input-ului utilizatorului din fereastra Mail Creator în format MIME, se utilizează o librărie externă (VMime). A fost aleasă o librărie externă, complexă, pentru că standardul MIME este în sine complex; s-a dorit o soluție implementată cât se poate de conformă standardului MIME și testată riguros.

3 Structura aplicației

3.1 Arhitectura principală

Proiectul folosește clase și OOP, având o structură inspirată de arhitectura MVP (Model-View-Presenter) pentru a separa responsabilitățile între clase.

3.2 Detalii despre arhitectură

Principala clasă care face legătura între interfețele aplicației și logica din spate (comunicarea cu servere, cu baza de date, etc.) este AppController. Aceasta administrează și comunică cu instanțe ale interfețelor LoginFrame, DashboardFrame și MailCreatorFrame și instanțe ale claselor Session și MailBuilder (utilizat când MailCreatorFrame este deschis). Clasa AppController poate fi privită ca un mediator între interfețe și logica programului, delegând operațiuni claselor specializate și trimițându-le interfețelor.

Session este clasa care se ocupă de comunicarea cu resurse externe pentru a obține și furniza informații. Aceasta solicită și trimite informații către instanțe ale altor clase precum FileManager, POP3Connection, SMTPConnection și DatabaseConnection, care realizează operații precum: crearea unui mailbox (sau potrivirea cu un mailbox deja existent) pentru utilizatorul conectat, obținerea mailurilor noi, salvarea lor în baza de date locală și în fișierele corespunzătoare, trimiterea unui mail nou, sau salvarea locală a unui atașament dintr-un mail.

În momentul primirii de notificări de la interfețele la care AppController este "abonat", AppController se ocupă de interacțiunea cu clasa Session pentru a obține informații și a le transmite către interfețe, care trebuie să folosească datele pentru a se actualiza și a le afișa către utilizator.

Datele care aparțin unui mail sunt păstrate la runtime cu ajutorul clasei Mail, care conține metode prin care se poate procesa conținutul unui buffer ASCII (scris conform standardului MIME) și se pot obține informații precum headers, conținutul plain-text sau HTML, atașamente, etc. În momentul când se dorește afișarea conținutului mailului pe ecran, interfața DashboardFrame este responsabilă cu afișarea datelor mailului în viewMailPanel, utilizând elemente specializate ale wxWidgets precum wxTextCtrl pentru plain-text sau wxHTMLWindow pentru conținut HTML cu imagini inline.

3.3 Stocarea informațiilor

La momentul deschiderii programului, în același folder cu executabilul se crează folderul `.poshet`, unde sunt create folderul `mail` (unde sunt stocate conținuturile "raw" ale mailurilor, în fișiere individuale) și baza de date SQLite `mail.db`, în care sunt stocate informații despre utilizatorul căruia îi aparține un mail, filename-ul unde poate fi găsit în folderul amintit mai sus, sau tag-ul care i-a fost asignat de către utilizator mailului respectiv.

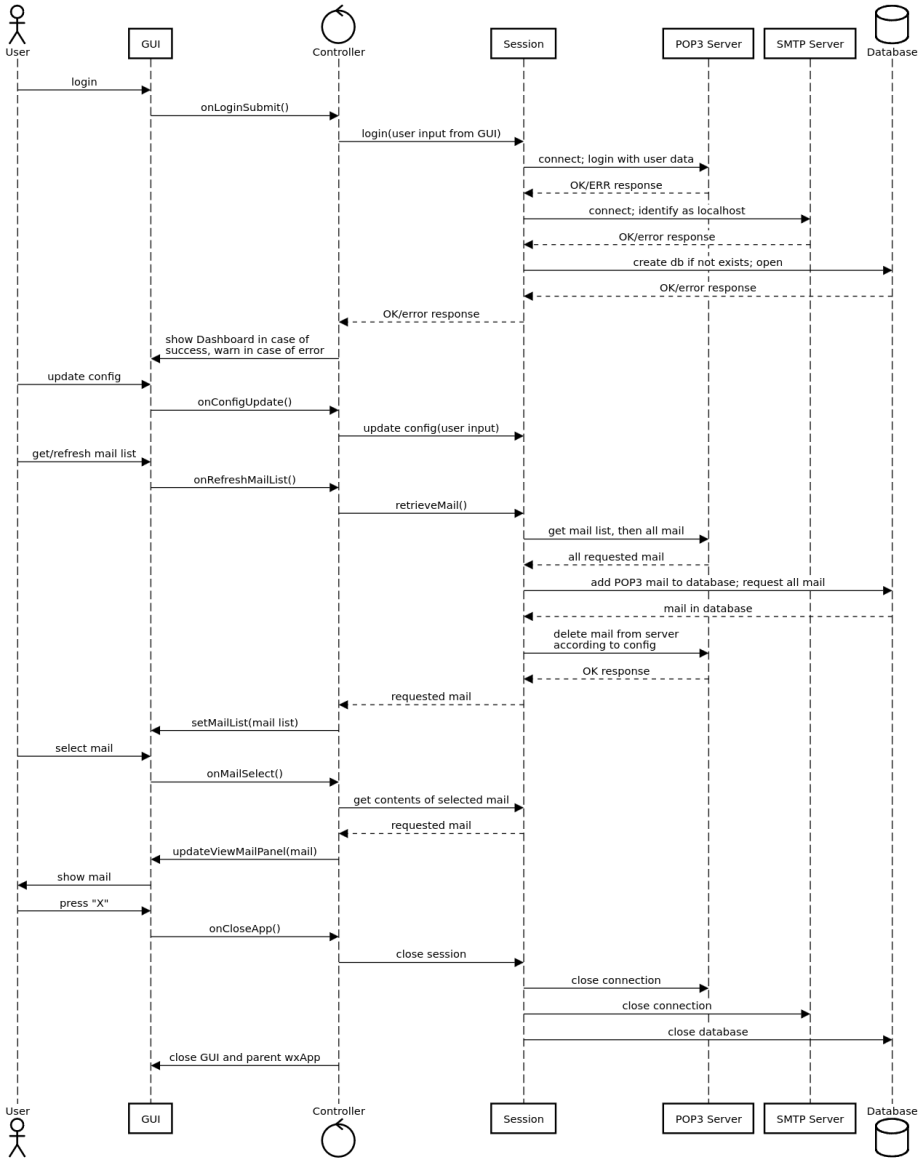

Figura 1. Diagrama secvențială a aplicației, partea 1: aici sunt ilustrate mecanismele de logare, afișare a mailurilor primite, și închidere a aplicației.

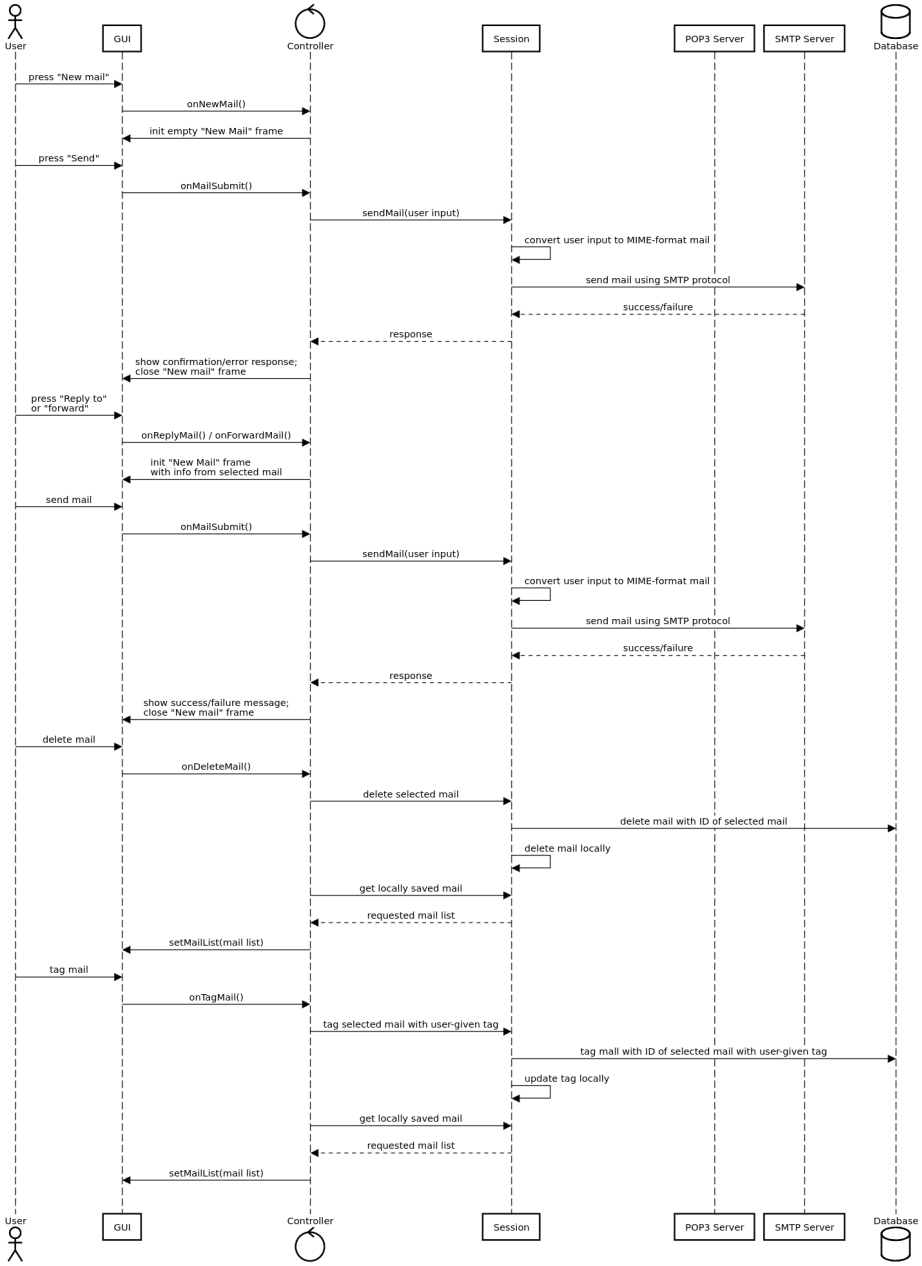

Figura 2. Diagrama secvențială a aplicației, partea 2: aici sunt ilustrate mecanismele de trimitere de mail nou și organizare de mail primit.

4 Aspecte de implementare

4.1 Protocoale utilizate

Proiectul constă într-un client pentru servere existente, deci acesta se folosește de protocoale la nivel de aplicație deja existente.

Pentru obținerea mailurilor se folosește protocolul POP3[1], iar pentru trimiterea de mailuri se folosește protocolul SMTP[2]. Clasele responsabile de conexiunile POP3 și SMTP trimit comenzi corespunzătoare funcționalităților dorite de utilizator, recunoscute de servere; spre exemplu, `USER / PASS` pentru autentificare POP3, `RETR` pentru lista de mailuri, `LIST` pentru datele unui mail particular, `MAIL TO` și altele pentru trimiterea unui mail, etc.

Ambele protocoale sunt de tip comandă-răspuns; așadar, în urma trimiterii comenzii se așteaptă răspuns afirmativ (sau un mesaj de eroare) de la server, informația fiind apoi transmisă la funcțiile care au solicitat-o.

4.2 Scenarii de utilizare

Poshet este menit să comunice cu un server POP3 (printr-o conexiune ori neîncriptată, ori încriptată prin SSL, cu autentificare plain-text prin comenzile `USER` și `PASS`), și cu un server SMTP (cu SSL sau nu, cu autentificare de tip `AUTH LOGIN` sau fără autentificare). Proiectul a fost testat atât pe mașina locală, cu adrese `@localhost` (cu un server Dovecot atât cu SSL, cât și fără, și un server Postfix, fără SSL, fără autentificare), dar și cu o adresă personală Yahoo (cu serverele `pop.mail.yahoo.com:995` și `smtp.mail.yahoo.com:465`).

Utilizatorul se poate loga cu adresa de e-mail și parola, în urma cărora aplicația va încerca să intuiască parametri precum "domain name"-ul serverelor POP3 și SMTP, sau username-ul pentru mailbox-ul POP3. Dacă autentificarea sau conectarea eșuează, există posibilitatea de a le introduce manual, prin field-urile marcate cu "(Optional)".

Ulterior, va fi afișată fereastra de tip "dashboard", care va afișa lista curentă de mailuri. Dacă este selectat vreunul, se afișează conținutul și detaliile acestuia în partea dreaptă a ecranului. În partea stângă a ecranului se regăsesc și alte elemente, precum butonul `Refresh mail list`, butonul `New message` prin care este inițializat procesul de trimitere a unui mail nou, sau o listă de directoare în care pot fi organizate mailurile din inbox.

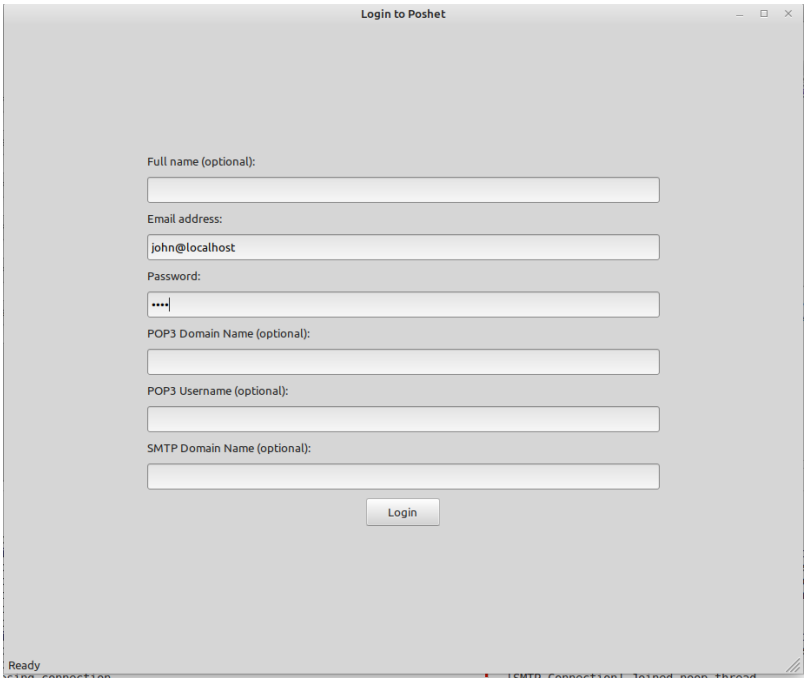

Figura 3. Fereastra "Login to Poshet".

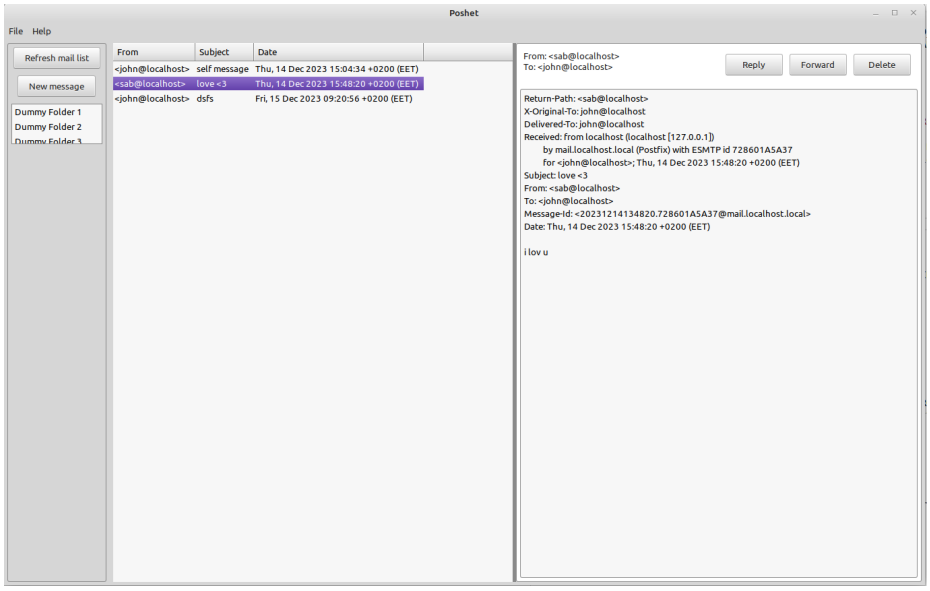

Figura 4. Fereastra de tip "dashboard", "Poshet". Aici, aceasta afișează conținutul "raw", neprocesat, al mailului selectat.

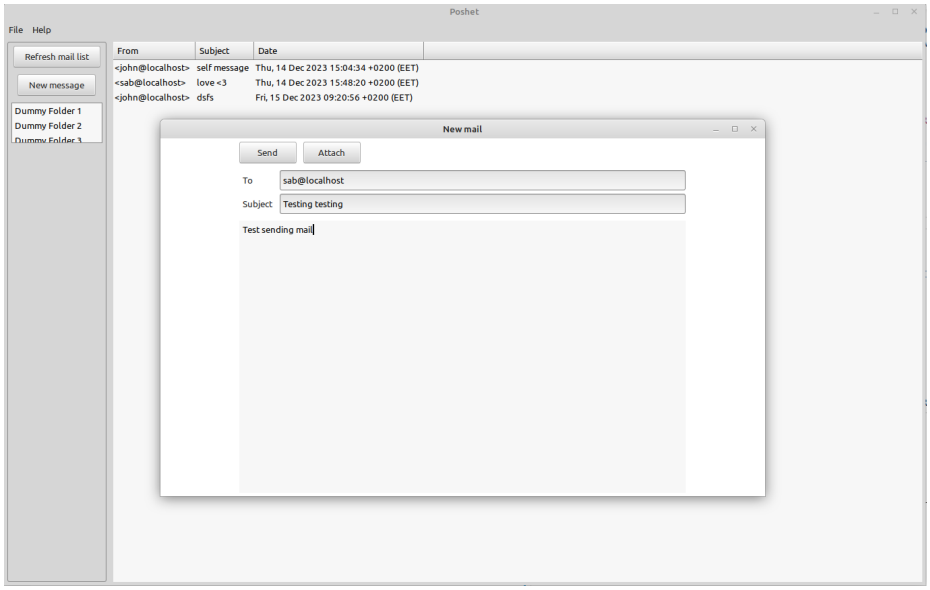

Figura 5. Fereastra de tip "New mail".

4.3 Aspecte interesante legate de implementare

Administrarea conexiunilor POP3 și SMTP. Conexiunile cu serverele POP3 și SMTP sunt administrate de clasele POP3Connection și SMTPConnection, care asigură comunicarea corectă cu acestea conform protocoalelor corespunzătoare.

În general, după o anumită perioadă de inactivitate configurată de către administratorii serverelor, acestea pot încheia conexiunea. Acest fapt nu este o problemă pentru SMTPConnection, fiindcă o conexiune SMTP nu trebuie să fie persistentă. În program, o conexiune SMTP este deschisă doar la logare (pentru a verifica corectitudinea datelor de logare) și în momentul în care este necesară trimiterea unui mail, fiind închisă imediat după. În schimb, conexiunea POP3 a fost proiectată în așa fel încât să poată fi persistentă, acomodând pentru un timp mai lung de procesare a unui mail după obținerea sa de la server. O particularitate interesantă a clasei POP3Connection este folosirea unui `std::thread` pentru a menține conexiunea activă: acesta trimite la un anumit interval repetat de timp, conform protocolului POP3, o operație de tip "no-op", prin care să semnaleze activitate fără a realiza o operație propriu-zisă.

În POP3Connection există membrul `std::thread _noopThread`, și o metodă `keepAlive()` care este executată de către acest thread. Pentru ca "no-op"-urile trimise de acest thread să nu interfereze cu procesul comandă-răspuns se folosește un `_commandMutex`, folosit în funcția `execCommand` care este apelată atât de comenzi reale, cât și de comanda "no-op". De asemenea, mutexul `_stateMutex`

este utilizat, printre altele, pentru a actualiza și verifica în mod sigur statusul curent al conexiunii.

Dacă apare o eroare (închiderea conexiunii sau time-out) în timp ce se dorește trimiterea unui "no-op", thread-ul își încheie execuția. Thread-ul principal va reverifica existența unei conexiuni active când va încerca să comunice la rândul ei cu serverul POP3, și va încerca reconexiunea.

```
void POP3Connection::keepAlive() {
    while(true) {
        {
            std::unique_lock<std::mutex> lock(_stateMutex);
            auto result = cv.wait_for(lock, std::chrono::seconds(TIMEOUT_SECS), [this]() {return _state != State::TRANSACTION;});
            if (result == true) {
                break;
            }
        }
        execCommand("NOOP");
        log("Sent NOOP");
    }
}
```

Figura 6. Definiția `keepAlive()` în `POP3Connection.cpp`.

5 Concluzii

Clientul Poshet este în principal single-threaded, folosind mai multe fire de execuție doar pentru a menține activă conexiunea POP3. O îmbunătățire posibilă ar fi adăugarea mai multor fire de execuție și modularizarea codului pentru a permite încărcarea mailurilor din inboxul POP3 "in the background" (pentru ca interfața să nu apară ca fiind "not responding" în timpul acestui proces), dar și pentru a putea întrerupe procesul dacă se dorește acest lucru.

De asemenea, librăria wxWidgets are propriile limitări. Comparativ cu alte librării precum Qt, widget-ul `wxRichTextCtrl` nu permite importarea fișierelor HTML, ceea ce limitează anumite abilități precum "Reply" și "Forward" la a funcționa în format plaintext. O îmbunătățire ar putea fi portarea proiectului la o altă librărie grafică cu un editor HTML de tip WYSIWYG ("what you see is what you get") precum Qt, sau dezvoltarea unui propriu "handler" bazat pe wxWidgets care să permită importarea HTML.

Bibliografie

1. RFC 1939, https://www.ietf.org/rfc/rfc1939.txt. Accesat cel mai recent pe 14 Dec 2023
2. RFC 5321, https://www.ietf.org/rfc/rfc5321.txt. Accesat cel mai recent pe 14 Dec 2023
3. RFC 2045, https://datatracker.ietf.org/doc/html/rfc2045. Accesat cel mai recent pe 14 Dec 2023
4. Pagina principală wxWidgets, https://www.wxwidgets.org/. Accesat cel mai recent pe 14 Dec 2023