

PROIECT RC-P

APLICAȚIE PENTRU TRANSFERUL DE FIȘIERE FOLOSIND PROTOCOLUL CU FEREASTRĂ GLISANTĂ

Budeanu Radu - Andrei
Cojocaru Constantin - Cosmin

Contribuții aduse:

Cojocaru Constantin-Cosmin:

- 2.1. Descrierea protocolului
- 2.2. Implementarea prin metoda Go Back N
- 3.4. Structura aplicației

Budeanu Radu Andrei:

- 2.3. Implementarea prin metoda selectarii repetitive
- 3.1. Metoda de implementare a protocolului cu fereastra glisanta
- 3.2. Standardul folosit pentru conexiunea dintre noduri
- 3.3. Limbajul de programare cu în care se va realiza aplicația și folosi bibliotecile aferente
- 3.4. Structura aplicației

Cuprins

	Pagina
1. Context.....	4
2. Protocolul cu fereastra glisanta (Sliding window protocol).....	5
2.1. Descrierea protocolului.....	5
2.2. Implementarea prin metoda Go Back N.....	6
2.3. Implementarea prin metoda selectarii repetitive (Selective Repeat).....	7
3. Descrierea aplicației.....	8
3.1. Metoda de implementare a protocolului cu fereastră glisantă.....	8
3.2. Standardul folosit pentru conexiunea dintre noduri.....	8
3.3. Limbajul de programare cu în care se va realiza aplicația și folosi bibliotecile aferente....	9
3.4. Structura aplicației.....	9
3.4.1. Interfața aplicației.....	9
3.4.2. Funcționalitatea aplicației.....	9

1. Context

Cerințe

Sa se implementeze o aplicație pentru transferul de fișiere între două noduri de rețea folosind protocolul cu fereastra glisanta.

Constrangeri:

- Două view-uri: unul pentru transmisie, unul pentru recepție
- Două sau mai multe perechi de instanțe ale aplicației (tx-rx) trebuie să poată să ruleze pe aceleași mașini sau în același LAN
- Comunicația va fi implementată prin datagrame UDP
- Formatul pachetelor va fi stabilit de echipe
- View-ul pentru recepție să permită „pierderea” voită a unor pachete pentru a putea demonstra funcționarea mecanismului
- Posibilitatea de configurare a parametrilor de funcționare (dimensiune fereastră, temporizări...)
- Se va alege una dintre următoarele variante de implementare: Go back n, Selective repeat fără Nack, Selective repeat cu Nack, implementarea TCP pentru controlul fluxului

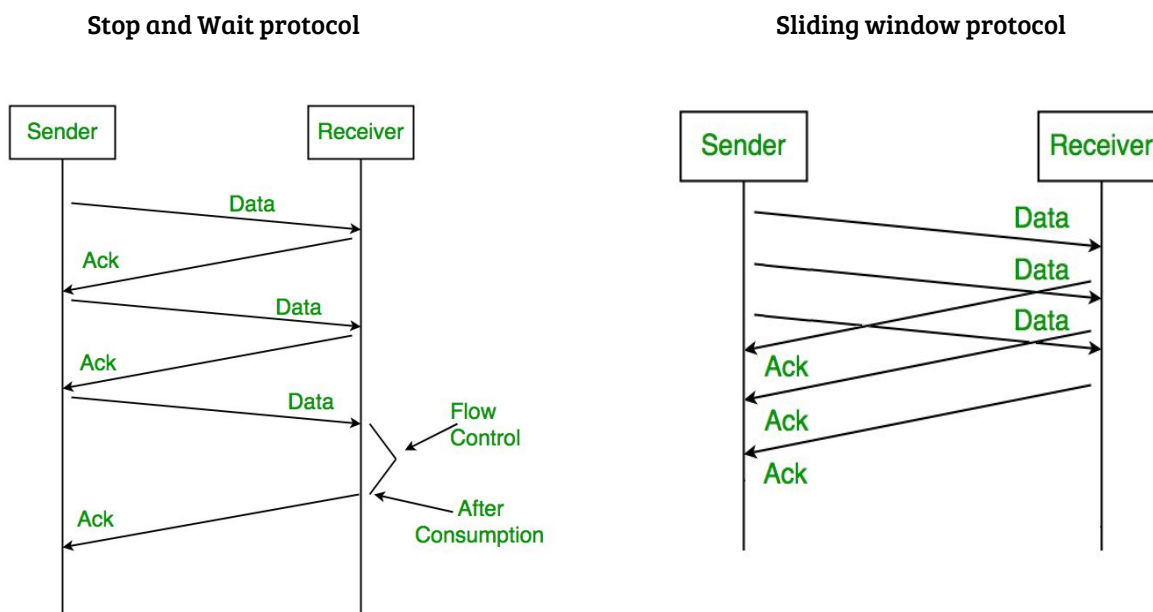
2. Protocolul cu fereastră glisantă (Sliding Window Protocol)

2.1. Descrierea protocolului

Protocolul cu fereastra glisanta este un protocol al nivelului legatura de date care facilitează transmiterea secvențială și mai eficienta a pachetelor de date de la un sender la un receiver. Acest protocol vine ca o îmbunătățire al protocolului Stop and Wait.

Principiul de funcționare al protocolului Stop and Wait: sender-ul trimite un pachet receiver-ului, urmand ca pachetul următor sa fie transmis doar din momentul în care se primește mesajul de confirmare al primirii pachetului.

Imbunatatirea adusa de protocolul cu fereastra glisanta este dat de crearea unui buffer de dimensiune N pachete la nivelul sender-ului (fereastra) care se trimit secvențial, secvențial urmând să se trimită și semnalele de confirmare.



Sursa imagini: GeeksforGeeks

Pentru implementarea acestui protocol exista mai multe implementări și vor fi prezentate după cum urmează: Go Back N, Selective repeat fără Nack, Selective repeat cu Nack.

2.2. Implementarea prin metoda Go Back N

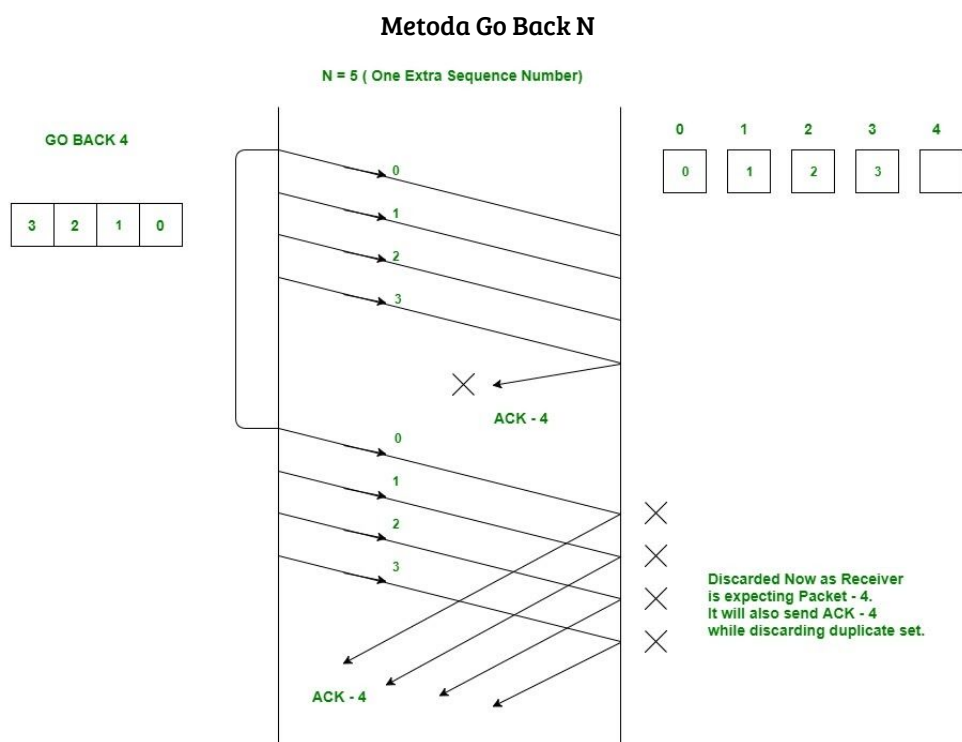
Metoda de implementare Go Back N este cea mai simpla metoda de implementare dintre cele enumerate.

La nivelul sender-ului este implementat un timer care reține timpul de la ultimul semnal de confirmare primit, iar fereastra acestuia este de mărime N pachete. Sender-ul va trimite secvențial pachetele din fereastra până la pachetul N.

La nivelul sender-ului avem două cazuri: primirea sau neprimirea semnalului de confirmare. La primirea semnalului de confirmare al primului pachet din fereastra, fereastra va face slide cu o poziție. Există însă posibilitatea ca primele k pachete să fie primite, dar pentru acestea se pierd semnalele de confirmare, însă sender-ul să primească semnalul de confirmare de la pachetul $k + 1$. În acest caz fereastra va face slide peste toate pozițiile pachetelor pentru care nu s-a primit confirmarea inclusiv peste poziția pachetului $k + 1$. Acest lucru se întâmplă deoarece primirea unui semnal de confirmare mai mare decât poziția actuală a ferestrei semnifică faptul că acele pachete k au fost recepționate de receiver. În cazul în care sender-ul nu primește semnal de confirmare după un anumit timp (timp de timeout), acesta va retransmite întreaga fereastră.

La nivelul receiver-ului avem un buffer de intrare de dimensiune 1. Receiver-ul trebuie să primească secvențial pachetele pentru a fi trimise mai departe către aplicație.

La nivelul receiver-ului, avem două posibilități: primirea următorului pachet sau primirea unui alt pachet care este după pachetul care ar trebui să fie primit. La primirea pachetului, receiver-ul verifică dacă acesta este pachetul următor ultimului pachet primit. Dacă acesta nu este următorul pachet, atunci receiver-ul va respinge acest pachet, dar și pachetele care mai vin. După un timp de timeout, receiver-ul va primi pachetul pe care îl așteaptă, retransmis de sender.



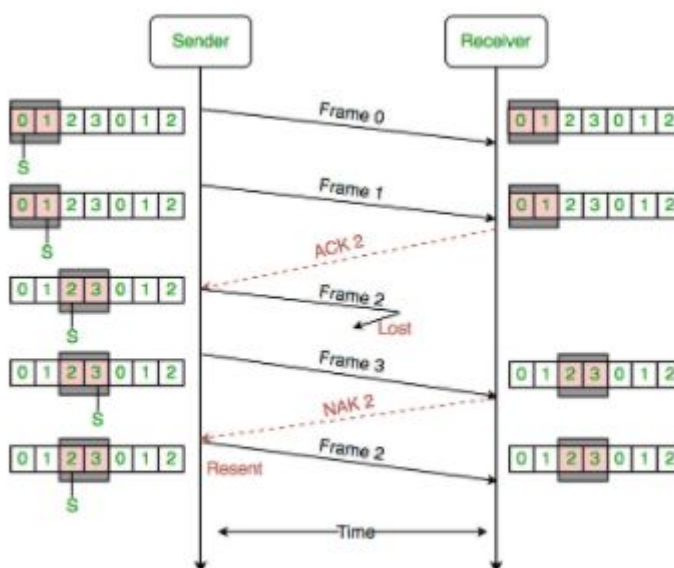
Sursa imaginii: GeeksforGeeks

2.3. Implementarea prin metoda selectari repetitive (Selective Repeat)

Dacă în timpul transmisiei pachetelor exista multe pierderi, algoritmul GO Back N devine foarte ineficient, utilizând prost lățimea de banda. Pentru a se veni cu o îmbunătățire, s-a dezvoltat sistemul cu selecție repetitivă, care de data aceasta va retransmite doar pachetele pierdute, care nu ajung la destinație, sau va retransmite pachetele ale căror confirmări nu au ajuns înapoi la sursă.

Spre deosebire de varianta GO Back N, acest sistem este mai greu de implementat, deoarece trebuie adăugată o anumită logică suplimentară pentru cele două noduri. Pentru nodul de transmisie, se poate opta pentru una dintre cele două implementări: selective repeat cu NACK sau selective repeat fără NACK. Pentru varianta cu NACK, retransmiterea pachetelor va fi cerută de către receptor printr-un mesaj de tipul “Not ACKnowledged” atunci când pachetele sunt eronate sau când se primește un pachet cu ordinea mai mare decât următorul din fereastra de recepție, iar transmițătorul va trebui să fie în stare să proceseze aceste mesaje și să transmită respectivele pachete. În varianta fără NACK, transmițătorul va trebui să măsoare timpul scurs de la transmiterea unui pachet. Dacă acest timp depășește o valoare de timeout și transmițătorul nu primește un mesaj de tipul ACK (acknowledge - ACK), atunci pachetul respectiv se va retransmite. Pentru receptor, va trebui implementată, de asemenea, o fereastră care este de aceeași dimensiune cu cea a transmițătorului, care va ordona pachetele pentru a fi salvate în ordinea corespunzătoare. În același timp, nodul care primește pachetele va trebui să genereze câte un semnal de validare pentru fiecare pachet primit și lipsit de erori de transmisie (dacă s-au putut detecta).

Selective Repeat cu NACK



Sursa imaginii: GeeksforGeeks

3. Descrierea aplicației

3.1. Metoda de implementare a protocolului cu fereastra glisanta

Pentru sistemul de transfer al fișierelor, se va implementa varianta cu selecție repetată (Selective Repeat) fără NACK. Asta înseamnă ca transmițătorul va avea cate un timer pentru fiecare pachet trimis din fereastra curentă, timer ce dacă va depăși valoarea de timeout stabilita pana la primirea unui semnal de validare, va trebui sa retransmită respectivul pachet.

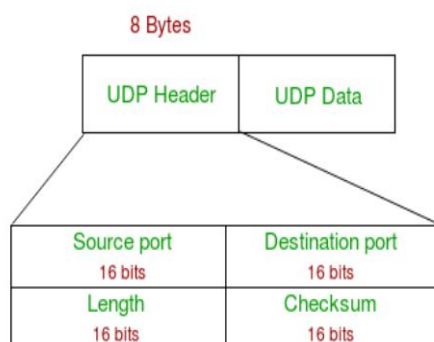
3.2. Standardul folosit pentru conexiunea dintre noduri

Aplicația va realiza conexiunea între sender si receiver folosind standardul UDP (User Datagram Protocol). Acest standard permite transferul rapid între noduri, fără a se stabili o “înțelegere” în prealabil (three hand shake-ul).

Deși nu este obligatoriu sa se implementeze o metoda de detecție a erorilor, deoarece standardul UDP ne permite acest lucru, vom realiza un sistem minimal de verificare a pachetelor folosind un checksum. Pentru fiecare pachet transmis, se va calcula valoarea sumei de control ca fiind un XOR pe 16 biti reprezentand cate 2 octeții ai datelor efective ce se doresc a fi transmise. Valoarea va fi salvată în header, iar la recepție, se va calcula din nou aceasta suma de control și se va realiza încă o operație XOR cu valoarea din header. Dacă rezultatul va fi diferit de 0, atunci pachetul este corupt și va fi nevoie sa fie retransmis.

Pachetul transmis prin intermediul standardului UDP arată astfel:

Formatul pachetului UDP



Sursa imaginii: GeeksforGeeks

În header se va defini port-ul sursa (care este opțional) și port-ul destinație, cat și lungimea în octeți a pachetului (incluzand header-ul) și valoarea sumei de control care este calculată după cum s-a explicat mai sus.

În campul pentru date se vor impacheta octetii generati de protocolul cu fereastra glisanta (continand și un header propriu).

3.3. Limbajul de programare cu în care se va realiza aplicația și folosi bibliotecile aferente

Pentru a implementa aplicația ce va transfera fișierele, se va folosi limbajul Python, utilizând doar biblioteca *sockets* (<https://docs.python.org/3/library/socket.html>) pentru conectarea la stiva de comunicare și respectiv între cele două noduri care comunică. Pentru diferitele operații matematice sau operații pe vectori se vor folosi librării precum *math*, *numpy*, iar pentru suportul de rulare paralelă se va folosi biblioteca *threading*. Pentru implementarea view-urilor se va folosi biblioteca PyQt.

Pentru implementarea aplicației se va folosi programare orientată pe obiecte.

3.4. Structura aplicației

3.4.1. Interfața aplicației

Fiecare instanță a aplicației va putea fi definită ca sender sau receiver din intermediul interfeței. Astfel, nu vor trebui să ruleze două aplicații diferite pe cele două mașini (de ex. *sender.py* și *receiver.py*). La nivelul sender-ului se va permite selectarea unui fișier pe care îl va transmite cât și dimensiunea ferestrei. La nivelul receiver-ului se va putea vizualiza pachetele recepționate.

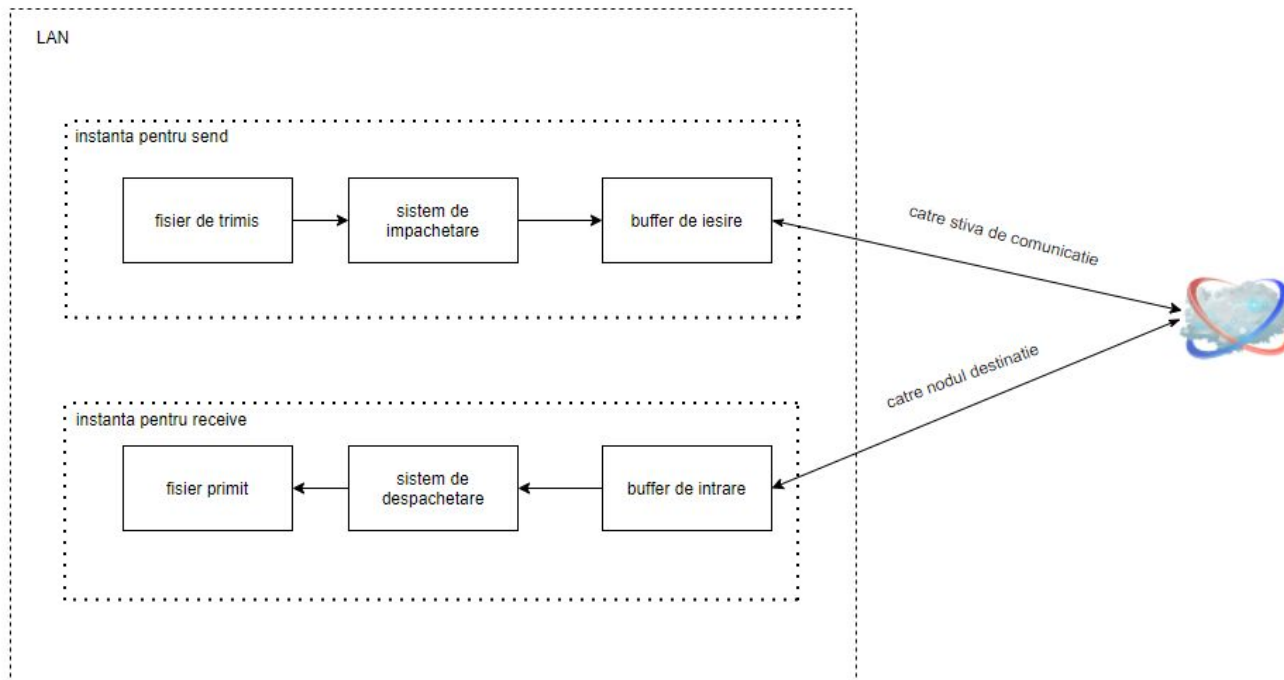
3.4.2. Funcționalitatea aplicației

Sender-ul va lua fișierul selectat, îl va citi și va crea pe rând pachetele conform unui standard definit, **(TO DO)** apoi le va trimite către stiva de comunicare prin intermediu UDP, setând și timer-ul pentru verificarea timeout-ului, în același timp. Sender-ul trebuie să fie capabil să proceseze la rândul lui mesaje de tip ACK în care va primi confirmarea că un pachet a ajuns cu succes. Ordinea de transmisie a pachetelor va fi determinată de o fereastră de lungime N, care prezintă funcționalitatea descrisă mai sus în documentație specifică pentru protocolul cu fereastră glisantă.

Receiver-ul va conține, de asemenea, o fereastră care determină ordinea de citire a pachetelor care sunt transmise. Acesta este capabil să transmită mesaje de tipul ACK atunci când pachetul este lipsit de erori. În cazul în care un pachet este eronat, atunci acesta nu va fi acceptat. Expirarea timeout-ului calculat de către sender va determina ca acel pachet să fie retransmis.

Fiecare nod al comunicării va conține câte un buffer de citire respectiv scriere, de unde se vor lua respectiv se vor pune pachetele.

Schema principală pentru flow-ul pachetelor



TO DO...