7. Nivelul transport

Nivelul transport este miezul întregii întregii ierarhii de protocoale, având ca sarcină transportul datelor de la sursă la destinaţie într-un mod sigur, eficace din puntul de vedere al costurilor şi independent de reţeaua fizică utilizată.
Nivelul transport este:

- Primul dintre nivelurile **de tip sursă-destinaţie [*end-to-end*]** - spre deosebire de primele trei, la care protocoalele se desfăşurau doar între două IMP de la capetele unui tronson de linie fizică dintr-o WAN);
- cel care separă nivelurile orientate pe aplicaţii (nivelurile 5, 6 şi 7 - *menite să asigure livrarea corectă a datelor între calculatoarele interlocutoare)*, de cele destinate operării subreţelei (nivelurile 1, 2 şi 3 - *responsabile cu vehicularea mesajelor prin reţea,* şi care pot suferi modificări de implementare fără a influenţa nivelurile superioare).

Nivelul 4 preia informaţia de la nivelul 5, o descompune, dacă e necesar, în unităţi mai mici (TPDU- Transport Protocol Data Unit = unitate de date a protocolului de transport), şi o trece nivelului 3, asigurând sosirea ei în formă corectă la destinatar.

7.1. Servicii

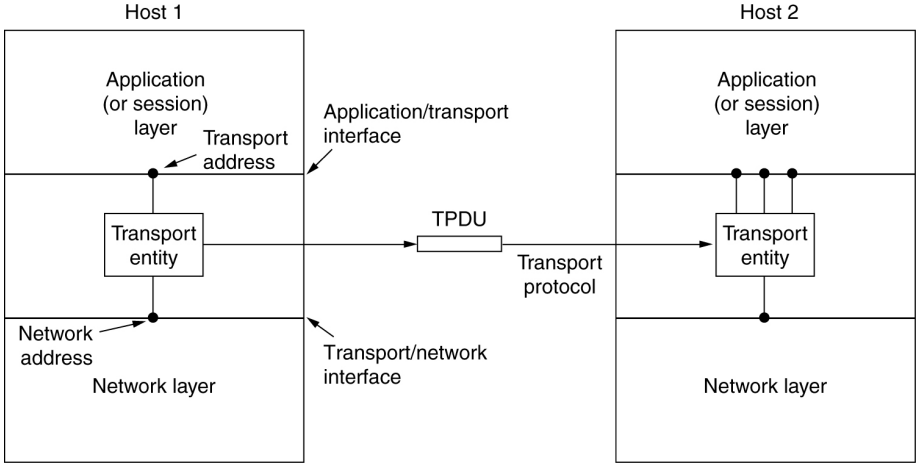
TCP asigură un serviciu orientat pe conexiune pentru transmisia fiabilă a datelor, cu detectarea erorilor şi controlul fluxului, şi deţine un mecanism de validare în 3 faze, acestea fiind:

- stabilirea conexiunii (handshaking)
- transferul datelor
- eliberarea conexiunii

Serviciile oferite de nivelul de transport nivelului 5 sunt de tipurile:

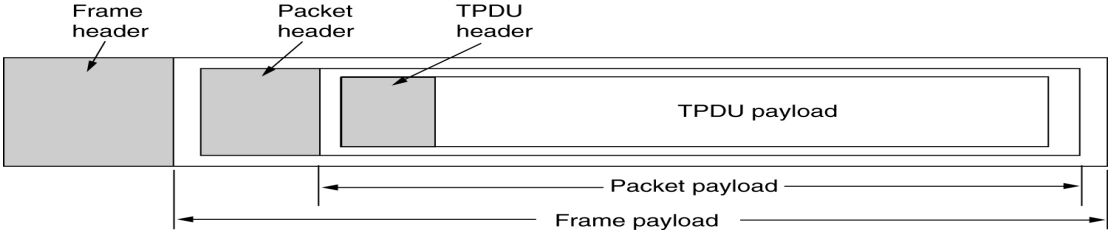
- o conexiune de transport de tip punct-la-punct, fără eroare, ce transmite mesajele în ordinea în care au fost emise;
- transportul unor mesaje izolate, fără garantarea ordinii la destinatar;
- difuzarea de mesaje către mai mulţi destinatari.

Relația logică între nivelurile rețea, transport şi aplicație este prezentată în continuare:



unde:

- *Entitatea de transport* - poate aparține nucleului SO, unui proces distinct, unei biblioteci legate de aplicațiile de rețea, sau se poate găsi în cadrul plăcii de rețea.
- *TPDU (Transport Protocol Data Unit)* – unitate de date a protocolului de date- reprezintă toate mesajele schimbate între 2 entități de transport corespondente. TPDU-urile de la nivel transport sunt conținute în pachete, la nivelul rețea, apoi în cadre, la nivelul legăturii de date.



Ierarhia de cadre, pachete, TPDU-uri

7.2. Funcții

- **controlul fluxului**

Se realizează printr-un mecanism fiabil de transport de date între sursă şi destinație, împiedicând ca să se transmită mai multe date decât pot fi recepționate.
Asigura una din sarcinile nivelului transport, aceea de asigurare a integrității datelor.
Transportul fiabil între sursă şi destinație implică o sesiune de comunicație orientată pe conexiune, protocolul asigurând că:

- emițătorul primeşte o confirmare din partea receptorului;
- toate pachetele care nu primesc confirmare vor fi retransmise;
- la destinație, pachetele vor fi aranjate în ordinea transmisiei, indiferent de ordinea în care au ajuns;
- se evită congestiile, supraîncărcarea sau pierderea de date.

- **comunicații orientate pe conexiune**

Foloseşte procedeul numit „three-way-handshake” (stabilirea conexiunii în 3 etape, care constă în:

- Emițătorul stabileşte o sesiune orientată pe conexiune cu receptorul, trimițând un apel *SYN*, un pachet de date cu un număr de secvență *n* şi care conține un bit în header, ce indică faptul că secvența reprezintă o cerere de conexiune;
- Receptorul primeşte pachetul, înregistrează secvența *n,*
- Receptorul răspunde cu o confirmare (*ACK*nowledgement) *n+1* şi include secvența inițială proprie cu numărul *m*; (confirmarea cu numărul n+1 înseamnă că s-au primit toți B de date până la n şi se aşteaptă următoarea secvență n+1);

Altă metodă de transmitere fiabilă a datelor este **PAR** (Positive Acknowledgment and Retransmission), în care emițătorul trimite un pachet de date, porneşte un contor de timp şi aşteaptă o confirmare înainte de a emite următorul pachet. Dacă timpul scurs până la primirea confirmării depăşeşte valoarea specificată, emițătorul retransmite acel pachet de date şi reporneşte contorul de timp.

- **memorarea temporară (**buffering)

Când o stație recepționează pachete mai repede decât poate procesa, acestea sunt stocate într-o zonă de memorie tampon, numită *buffer*. Rezolvă problema traficului supraaglomerat doar dacă este de scurtă durată. Dacă supraaglomerarea persistă, memoria tampon va deveni insuficientă, iar pachetele se vor pierde.

- **transmisia de confirmări** (windowing)

Windowing – fereastra de date - procedeu de transmitere a datelor de la emițător la receptor, care ataşează confirmarea după mai multe pachete de date, şi nu după fiecare în parte (ceea de scade viteza de transmisie).
Metode de folosire a acestei tehnici:

- protocoale care cuantifică informația trimisă într-o fereastră de date ca multiplu de pachete;
- protocoale (inclusiv TCP) care folosesc multipli de B pentru ferestrele de date. Exp.: într-o fereastra cu dimensiunea 3, se vor transmite 3 pachete de date fără a aştepta confirmarea. TCP foloseşte tehnica de „fereastră glisanta" (sliding-windows), prin care dimensiunea ferestrei de date este negociată dinamic în timpul transmisiei.

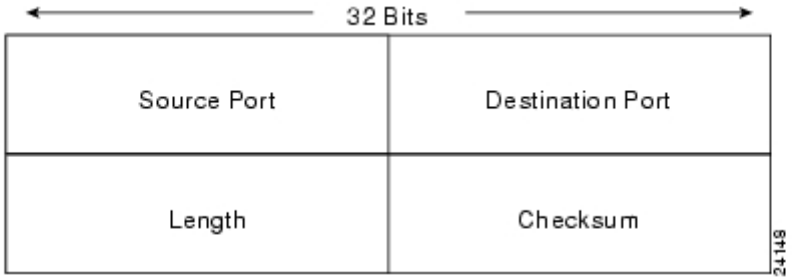
- stabilirea şi eliberarea conexiunii;
- multiplexarea
- refacerea după căderi
- adresarea - translatarea adresă transport-adresă rețea;
- transferul unităților de date normale şi speciale;
- numerotarea TPDU, secvențierea unităților de date;
- detectarea erorilor şi supravegherea calității serviciului;
- segmentarea, gruparea, concatenarea.

7.3. Protocoale de transport prin Internet

7.3.1. UDP

- descris în RFC 768

- utilizat ca multiplexor/demultiplexor pentru emiterea şi recepţionarea datagramelor
- pentru direcţionarea datagramelor foloseşte porturile;
- oferă un serviciu de transmisie a datagramelor fără conexiune, nefiabil, fără mecanism pentru controlul fluxului sau recuperarea erorilor, fără asigurarea în caz pe pierdere sau recepţionare în alta ordine;
- fiecare datagramă UDP este emisă într-o singură datagramă IP (care poate fi fragmentată pe durata transmisiei şi reasamblată pt. UDP)
- toate implementările IP acceptă datagrame de 576B, cu un antet de 60B => datagrama UDP = 516B;
- există implementări care acceptă şi datagrame mai mari, dar nu sunt garantate.
- Aplicaţiile standard care utilizează UDP sunt: TFTP (Trivial FTP), DNS, RPC, SNMP, LDAP (Lightweight Directory Access Protocol);
- Structura datagramei UDP:



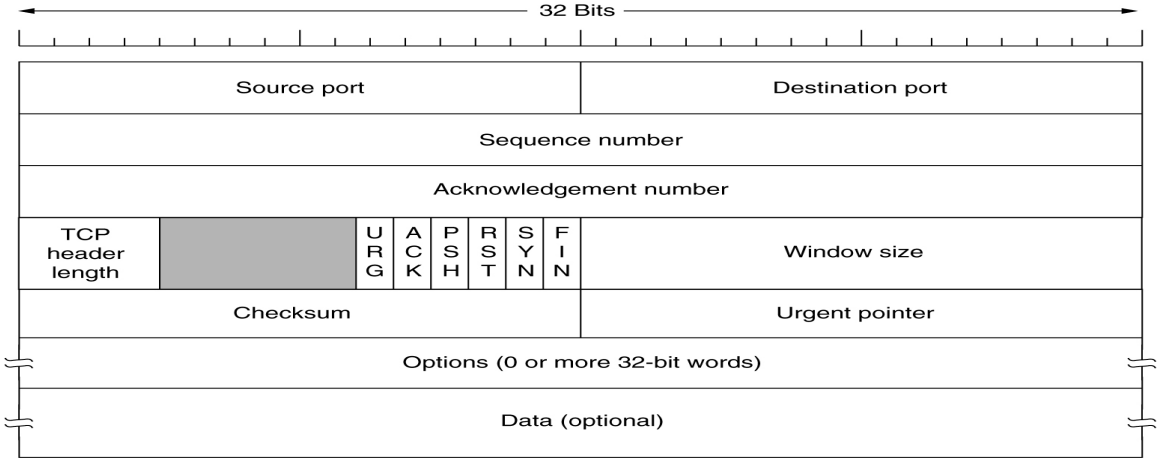
  - *Port sursă /destinaţie* (16b) – identifică numărul de port al procesului emiţător şi care va fi adresat în răspuns/ numărul de port al procesului destinaţie;
  - *Lungime* – lungimea datagramei, inclusiv antetul;
  - *Suma de control* (16b) – este opţională, se calculează pentru pseudoantetul IP, antetul UDP, datele UDP. Pseudoantetul UDP conţine: adresele IP sursă şi destinaţie, protocolul, lungimea UDP;
  - *Datele* – urmează antetului UDP.

7.3.2. TCP

- RFC 793
- Asigură un *serviciu orientat pe conexiune* pentru:
  - *transmisia fluxului de date* – transferă un flux continuu de B direct în reţea, fără să se ocupe de divizarea în blocuri sau datagrame; B sunt grupaţi în segmente TCP, încapsulate în datagrame IP;
  - *fiabilitatea conexiunii* – asociază un număr de secvenţă fiecărui B transmis şi aşteaptă o confirmare (ACK –acknowledgment) de la receptor; dacă nu primeşte confirmarea după un anumit interval de timp, va retrimite datele (doar numărul de secvenţă al primului B din segment); numerele de secvenţă se folosesc pentru ordonarea segmentelor şi eliminarea duplicatelor;
  - *detectarea erorilor*,
  - *controlul fluxului* – când primeşte un ACK, specifică emiţătorului numărul de secvenţă al următorului B pe care-l aşteaptă; pentru mărirea debitului se foloseşte mecanismul de *fereastră glisantă*;
  - *multiplexar*e- realizată prin utilizarea porturilor;

- *Serviciu duplex* – realizează transmisii ale fluxului datelor în ambele direcții, în același timp;
- *Conexiune logică* – reprezintă starea fluxului de date; este identificată unic printr-o pereche de socket-uri ale emițătorului şi receptorului; este determinată de următoarele elemente:
  - Socket-uri – asocierea dintre o adresă IP şi un număr de port;
  - Numere de secvență;
  - Mărimea ferestrei;
- mecanism de validare în 3 faze, care constă din:
  - *Stabilirea conexiunii* – presupune o negociere în 3 paşi:
    - TCP client solicită stabilirea unei conexiuni, emițând o cerere de sincronizare şi un număr inițial de secvență: SYN, n;
    - TCP server confirmă cererea de conexiune, cere clientului sincronizarea cu numărul său inițial de secvență: SYN, m, ACKn+1;
    - TCP client confirmă cererea de sincroniza: ACKm+1;
  - *Transferul datelor*;
  - *Eliberarea conexiunii*;

- Realizează un circuit logic fiabil între perechi de procese;
- Nu răspunde de fiabilitatea protocoalelor nivelurilor inferioare (exp.: IP);
- Este folosit de majoritatea protocoalelor, inclusiv de FTP, TELNET, HTTP.
- Formatul segmentului TCP este:



- *Port sursă/destinație* – (16b), identifică aplicația emițătoare/receptoare;
- *Număr de secvență* – (32b) numărul de secvență al primului B din segment; la stabilirea conexiunii se negociază valoarea inițială a numărului de secvență;
- *Număr de confirmare* – (32b) numărul de secvență pentru următorul B pe care-l aşteaptă receptorul;
- *Lungime antet* – (4b) numărul de cuvinte de 32b din antetul segmentului TCP;
- *Indicatori* – (6b)
  - *URG* =1 => pentru acest segment este important *indicatorul de urgență*;

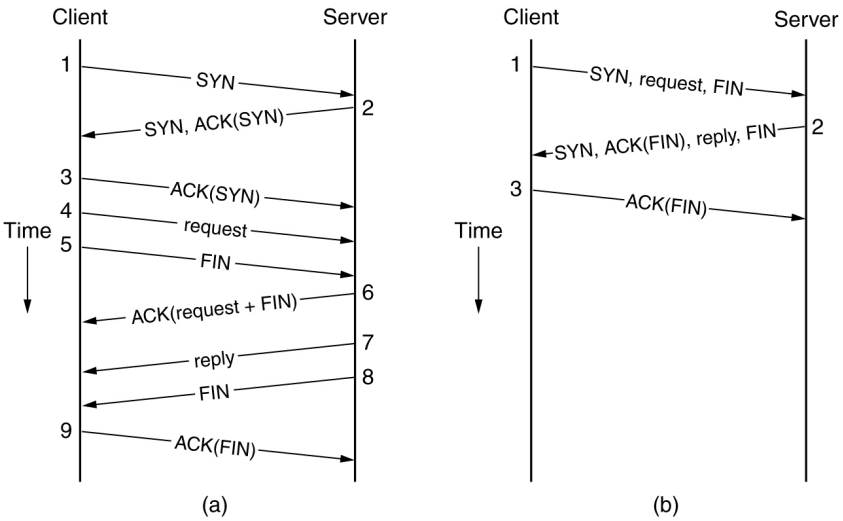
- *ACK* =1 => pentru acest segment este important *numărul de confirmare*;
- *PSH* =1 => (opțional) transmite imediat datele, fără memorare în tampoanele de comunicație;
- *RST* =1 => reinițializarea conexiunii TCP;
- *SYN* =1 => cerere conexiune (dacă ACK=0), conexiune acceptată (dacă ACK=1) + sincronizarea numărului de secvență;
- *FIN* =1 =>conexiunea se încheie, nu mai sunt date de transmis;

- *Dimensiune fereastră* – numărul de B care pot fi trimişi într-o fereastră glisantă;
- *Sumă de control* – se calculează pentru fiecare segment TCP, şi permite destinației să detecteze erorile;
- *Indicator de urgență* – specifică ultimul B de date urgente, dacă URG=1;
- *Opțiuni* – facilități care nu au fost introduse în antet (exp: dimensiunea maximă a segmentului).

7.3.3. TCP Tranzacțional

T/TCP - RFC 1379, 1644- variantă experimentală care combină eficiența RPC-ului bazat pe UDP cu fiabilitatea TCP-ului;

- modifică secvența standard de inițializare a conexiunii pentru a permite transferul de date în timpul inițializării;
- primul pachet al clientului conține bitul SYN, cererea şi FIN (vreau o conexiune, am datele, terminat);
- serverul primeşte cererea, determină răspunsul şi alege modul de răspuns;
- clientul confirmă cu FIN şi se termină protocolul în 3 mesaje.



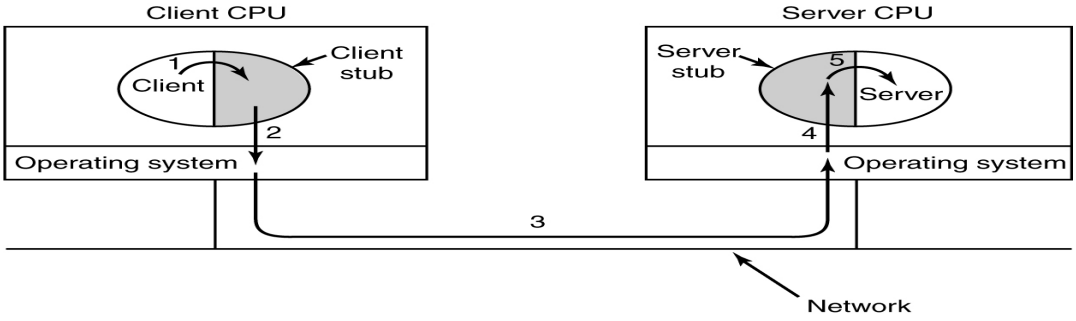
SCMP (Stream Control Transmissio Protocol) – protocol de control al transmisiei fluxului – este o îmbunătățire adusă lui TCP privind găzduirea multiplă, confirmări selective, modalități multiple de livrare, păstrarea legăturilor dintre mesaje.

7.4. Alte protocoale

7.4.1. RPC

RPC (Remote Procedure Call) – apel de procedură la distanță;

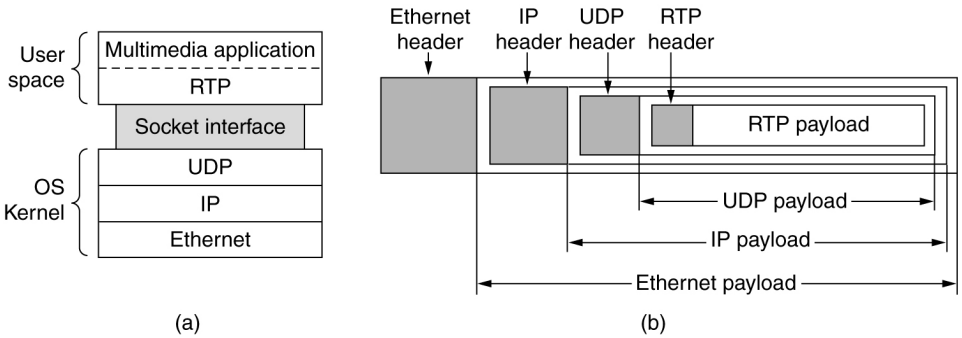
- este baza pentru aplicații de rețea: procedura care apelează este clientul, iar cea apelată este server-ul;
- programul client trebuie să fie legat de o mică procedură de bibliotecă, numită client stub (ciot), care reprezintă procedura server-ului în spațiul de adresă al clientului. Similar, serverul este legat cu server stub.
- Paşii realizării RPC:
    1. Clientul apelează stub-ul client (apel de procedură locală);
    2. Împachetarea parametrilor de către stub-ul client într-un mesaj şi realizarea unui apel de sistem pentru a trimite mesajul;
    3. Nucleul SO trimite un mesaj de la client la server;
    4. nucleul trimite pachetele sosite la stub-ul server;
    5. stub-ul server apelează procedura server cu parametrii despachetați.



7.4.2. RTP

RTP (Real-Time Transport Protocol)- protocol de transport în timp real, utilizat pe scară largă acum;

- RFC 1889;
- utilizat în cadrul aplicațiilor multimedia în timp real;
- are o poziție ciudată în stiva de protocoale: rulează peste UDP, în spațiul utilizator;
- este un protocol generic, independent de aplicație, care furnizează facilități de transport (este un protocol de transport implementat la nivel aplicație);
- funcționare:
    - aplicațiile multimedia sunt trimise bibliotecii RTP (aflată în spațiul utilizator, ca şi aplicația);
    - biblioteca multiplexează fluxurile, le codează în pachete RTP, le trimite printr-un soclu;
    - la celălalt capăt al soclului, în nucleul SO, pachetele UDP sunt generate şi încapsulate în pachete IP



a) Poziționarea RTP în stiva de protocoale b) încapsularea pachetului

7.4.3. RTCP

RTCP (Real-Time Control Protocol)- RFC 1890- este protocolul de control care lucreaza in conjunctie cu RTP, dar nu garanteaza QoS.

- Ofera suport pentru aplicatii multimedia pe grupuri mari din internet, incluzand identificarea sursei si suport pentru gateways si multicast-to-unicast translators.
- Se ocupă de răspuns, sincronizare, interfața cu utilizatorul, dar nu transportă date.
- este folosit pentru a strange informatii despre performanta conexiunii. Aceste informatii sunt folosite pentru ajustarea dinamica si optimizarea conditiilor de retea curente. Protocolul specifica raportul de pachete schimbate intre surse si destinatii ale informatiei multimedia .
- este o tehnologie care ajuta RTP sa ruleze mai efficient, mai ales in cazul unor conexiuni mai slabe ca viteza, prin compresia RTP/UDP/IP. Acest lucru este benefic in special pentru pachete mai mici (cum ar fi IP voice traffic) pe conexiuni mai slabe, unde compresia RTP header poate reduce intarzierea si overhead-ul semnificativ.

7.5. Adresarea aplicațiilor

Un calculator are in general o singura legatura fizica la retea. Orice informatie destinata unei anumite masini trebuie deci sa specifice obligatoriu adresa IP a acelei masini. Insa pe un calculator pot exista concurent mai multe procese care au stabilite conexiuni in retea, asteptand diverse informatii. Prin urmare datele trimise catre o destinatie trebuie sa specifice pe langa adresa IP a calculatorului si procesul catre care se indreapta informatiile respective.

Identificarea proceselor se realizeaza prin intermediul **porturilor**.

Un *port* este un numar pe 16 biti care identifica in mod unic procesele care ruleaza pe o anumita masina. Orice aplicatie care realizeaza o conexiune in retea va trebui sa ataseze un numar de port acelei conexiuni. Valorile pe care le poate lua un numar de port sunt cuprinse intre 0 si 65535 (deoarece sunt numere reprezentate pe 16 biti), numerele cuprinse intre 0 si 1023 fiind insa rezervate unor servicii sistem si, din acest motiv, nu trebuie folosite in aplicatii.

Exemple de porturi utilizate pentru serviciile Internet: ftp 21/tcp, telnet 23/tcp, netstat 15/udp, netstat 15/tcp etc.

Există numere de porturi care pot fi alocate dinamic.

Adresarea aplicațiilor este un exemplu de funcționare a *multiplexării*, putând exista mai multe conexiuni transport pentru o singură conexiune de rețea. Datele care provin de la nivelul rețea trebuie distribuite proceselor destinație, realizând *demultiplexarea*. Aceasta necesita ca IP să utilizeze:
- numere de protocol – pentru a determina protocolul de serviciu;
- numere de port – pentru a identifica serviciile.

Socket-ul, inovație a sistemului Berkeley Unix,
- este un punct de comunicație, prin care un proces poate emite sau recepționa informație sub forma unui flux de B.
- Este identificat asemănător fişierelor, printr-un descriptor.
- Adresa unui socket este tripletul: protocol, adresă_locală, proces_local.
- Interfața socket este un API pentru rețelele TCP/IP.

8. COMPLETARE - NIVELUL LEGĂTUR Ă DE DATE –

8.1. METODE DE ACCES

După tipul canalului, rețelele pot fi punct-la-punct şi cu difuzare.

Rețelele cu difuzare utilizează canale cu difuzare.

În rețelele cu difuzare se pune problema accesului la mediu pentru mai mulți utilizatori simultani.

Canalele pot fi: multiacces şi cu acces aleator

Protocoalele folosite pentru a stabili cine transmite într-un canal multiacces aparțin subnivelului MAC, important mai ales în LAN-uri.

Rețelele WAN utilizează legături punct-la-punct, cu excepția celor prin satelit.

Scheme de alocare pt. LAN-uri – protocoale cu acces multiplu:

- cele mai simple scheme de alocare: FDM, TDM – când numărul de stații este mic, traficul continuu
- ALOHA cu sau fără cuantificare – pt. număr mare de stații, variabil, trafic de tip rafală;
- Detectarea purtătoarei –când starea canalului poate fi detectată, iar stațiile pot evita începerea transmisiei, dacă există o altă stație care transmite;
- Cu conflict limitat - Parcurgere arborescentă –împarte dinamic stațiile în 2 grupuri disjuncte, unul având drept de transmitere, iar altul nu. Doar o stație va transmite, dintre cele pregătite de trimită;
- Acces multiplu şi detecție de purtătoare:
  - CSMA persistent,
  - nepersistent,
  - cu detecția coliziunii CSMA/CD (Carrier Sense Multiple Access with Collision Detection – 2 stații transmit simultan, are loc o coliziune, care este detectată urmărind puterea sau lățimea impulsului semnalului recepționat şi comparat cu cel transmis. După detectarea coliziunii, stația aşteaptă o cuantă de timp şi încearcă din nou să transmită. Vor fi 3 perioade timp: de conflict, de aşteptare, de transmisie.
- Fără coliziuni –
  - protocol bit-map (hartă de biți) şi
  - Numărare binară inversă – elimină complet conflictele;
- **pentru WLAN** (LAN fără fir):
  - MACA (Multiple Access with Collision Avoidance –acces multiplu cu evitarea coliziunii – pt IEEE 802.11- emițătorul stipulează receptorul să emită un cadru scurt, astfel stațiile apropiate să poată detecta această transmisie şi să nu emită pe perioada cadrului.
  - MACAW - MACA îmbunătățit, cu cadre de confirmare ACK după fiecare cadru de date transmis cu succes
  - Se mai utilizează şi **FHSS** (Frequency Hopping Spread Spectrum), **DSSS** (Direct Sequence Spread Spectrum)