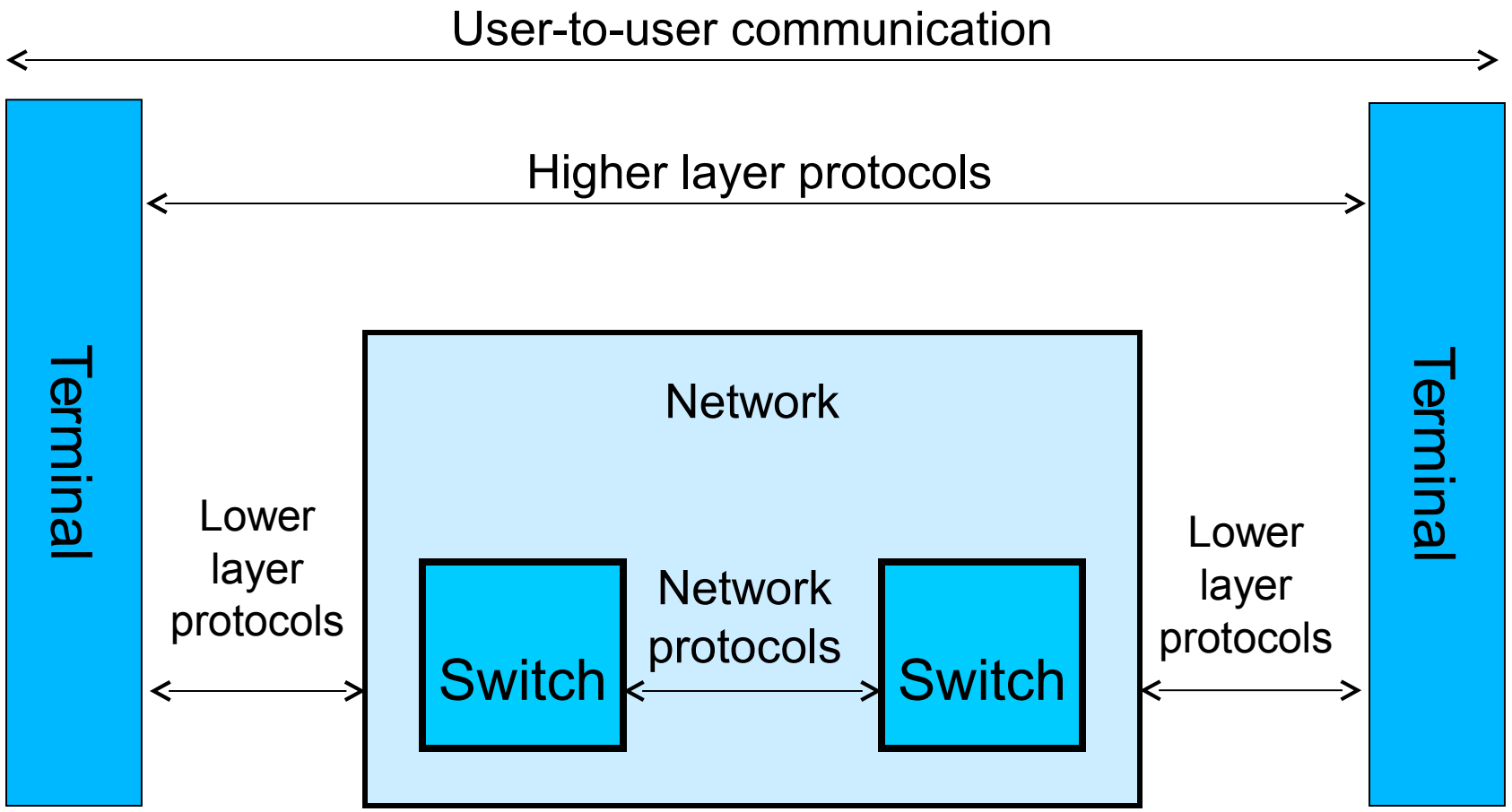
Simple communication model



SIEŤOVÝ SOFTWARE

Počítačové siete (PS) v minulosti sa navrhovali hlavne z pohľadu hardwaru, software až pre už navrhnutý hardware. Táto stratégia už nefunguje, sieťový software je vysoko štrukturovaný a v ďalšom sa budeme zaoberať technikou štrukturovania sieťového softwaru.

Za účelom zníženia zložitosti návrhu, väčšina PS je chápaná ako postupnosť sieťových vrstiev, úrovní (layers). Každá vrstva je vytvorená nad vrstvou, ktorá je pod ňou. Počet vrstiev, názov, obsah, funkcia vrstiev sa mení od siete k sieti.

Vo všetkých sieťach cieľom vrstvy je ponúknuť služby vyššej vrstve, pričom detaily ako sú služby implementované, sú pre vyššie vrstvy neznáme (zakryté).

N-tá vrstva na jednom počítači komunikuje (udržuje konverzáciu) s N-tou vrstvou na druhom počítači. Pravidlá a zásady použité na túto konverzáciu sa súhrne nazývajú ako ***protokol* pre N-tú vrstvu**.

Protokol je vo všeobecnosti dohoda medzi komunikujúcimi stranami o tom, ako má prebiehať komunikácia.
Obrázok nižšie ilustruje 5-vrstvovú sieť

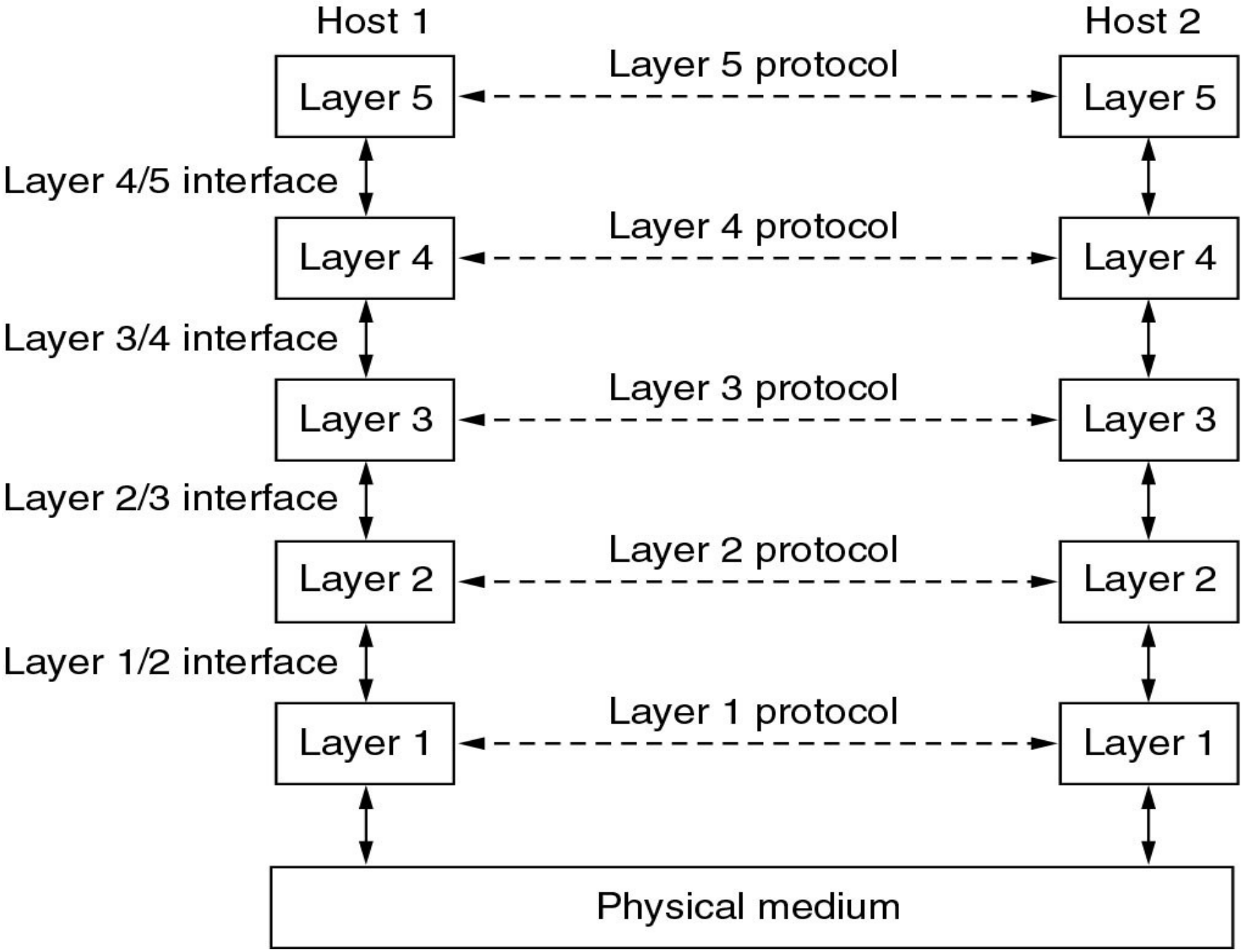
Peers (funkčné jednotky na rovnakej vrstve) - sú entity nachádzajúce sa v navzájom si zodpovedajúcich vrstvách na rôznych počítačoch (systémoch, uzloch), t.j. tie entity, ktoré komunikujú podľa protokolu danej vrstvy. Peers môžu byť procesy, hardwarové zariadenia, čipy, ...

V skutočnosti (!) žiadne údaje sa neposielajú priamo z N-tej vrstvy na jednom počítači do N-tej vrstvy na druhom počítači. Ale každá vrstva pošle údaje a riadiacu informáciu cez vrstvu bezprostredne pod ňou..

Nakoniec sa dosiahne prvá vrstva, pod ktorou sa nachádza fyzické prenosové médium, cez ktoré sa komunikuje s druhou stranou.

virtuálna komunikácia - čiarkované šípky na obrázku
fyzická komunikácia - plné šípky na obrázku

Interface medzi vrstvami - definuje, ktoré základné operácie a služby nižšia vrstva ponúka vyššej.


Host 1
Host 2
Layer 5
Layer 5 protocol
Layer 5
Layer 4/5 interface
Layer 4
Layer 4 protocol
Layer 4
Layer 3/4 interface
Layer 3
Layer 3 protocol
Layer 3
Layer 2/3 interface
Layer 2
Layer 2 protocol
Layer 2
Layer 1/2 interface
Layer 1
Layer 1 protocol
Layer 1
Physical medium

Pri rozhodovaní koľko vrstiev navrhnúť je najpodstatnejším kritériom jednoduchý a jasný interface medzi vrstvami. To znamená, že každá vrstva vykonáva špecifickú skupinu dobre zrozumiteľných funkcií. Okrem minimalizácie množstva prenosu informácie medzi vrstvami má jednoduchý a jasný interface umožniť napr. aj náhradu starej implementácie jednej vrstvy úplne novou odlišnou implementáciou tej istej vrstvy. Od novej implementácie sa len požaduje, aby ponúkala tie isté služby pre vyššiu vrstvu ako stará implementácia.

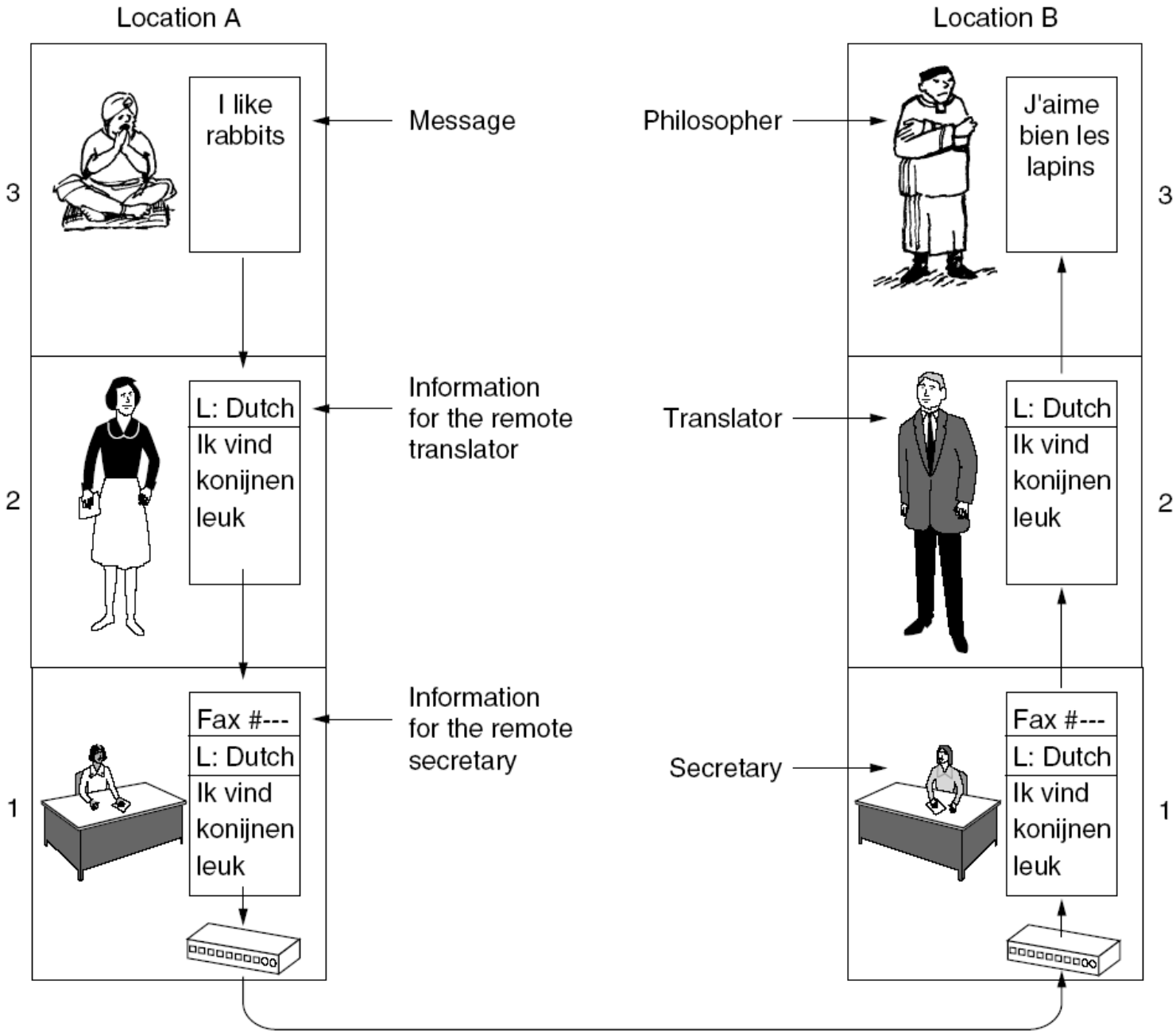
Sieťová architektúra (network architecture) - množina vrstiev a protokolov. Špecifikácia architektúry musí obsahovať dosť informácie na napísanie programu resp. zhotovenie hardwaru pre každú vrstvu tak, aby spĺňal protokol danej vrstvy.

Naproti tomu, detaily implementácie, ani špecifikácia interfacu nie su časťou architektúry.

Dokonca nie je nutné, aby interfacy na všetkých počítačoch v sieti boli rovnaké, za predpokladu, že každý počítač korektne používa všetky protokoly.

Zásobník protokolov (protocol stack) - zoznam protokolov používaný určitým systémom pri konkrétnej komunikácii. Pre každú vrstvu je do zásbníka (zoznamu) vybraný práve používaný protokol.

Ideu viacvrstvovej komunikácie vysvetľuje obrázok o komunikácii dvoch filozofov.


Location A
Location B
I like rabbits
Message
Philosopher
J'aime bien les lapins
3
3
L: Dutch
Ik vind konijnen leuk
Information for the remote translator
Translator
L: Dutch
Ik vind konijnen leuk
2
2
Fax #---
L: Dutch
Ik vind konijnen leuk
Information for the remote secretary
Secretary
Fax #---
L: Dutch
Ik vind konijnen leuk
1
1

Poznámka:

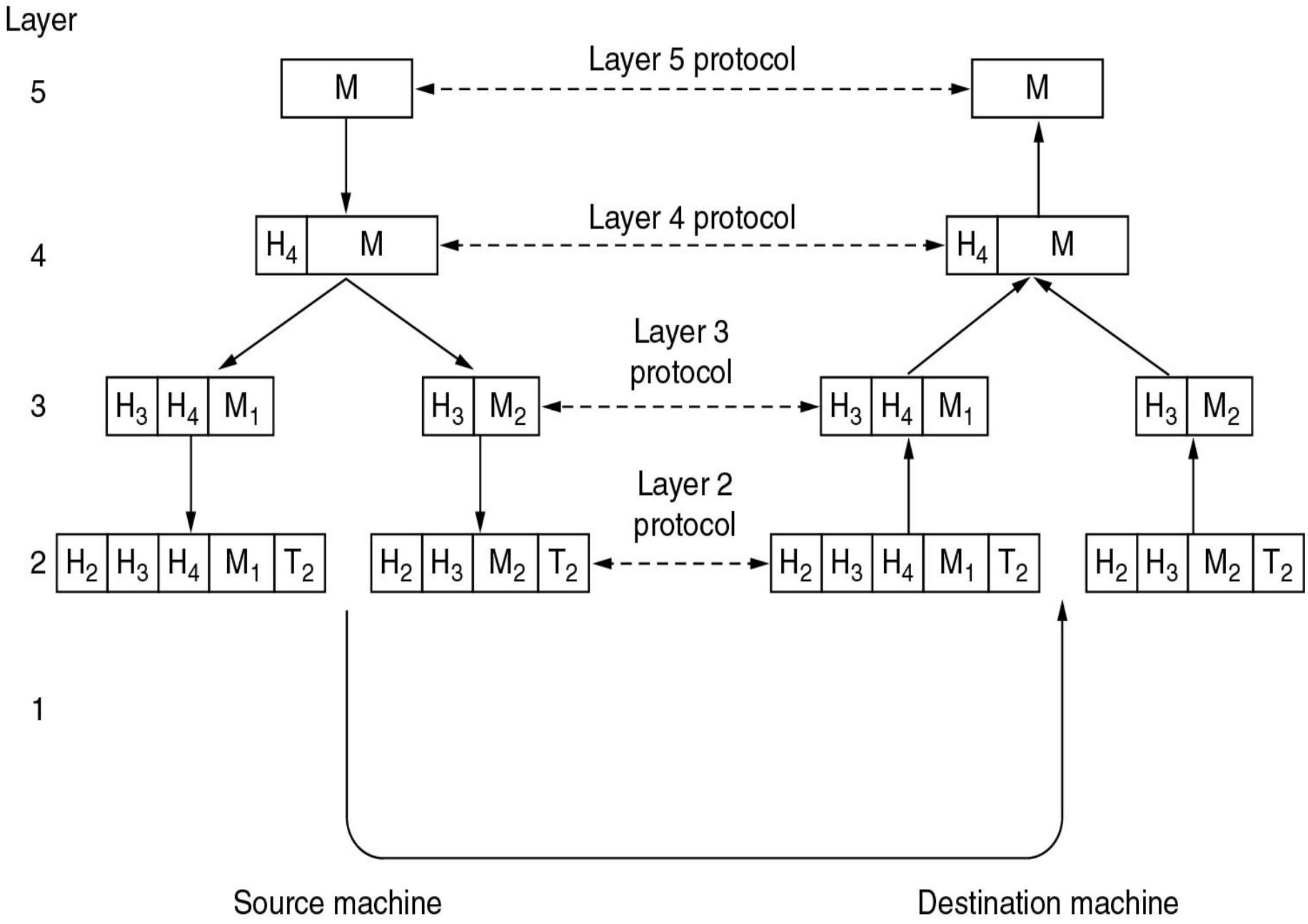
Každý protokol je nezávislý od ostatných, pokiaľ sa nemení interface.

Napr. prekladatelia môžu prepnúť do iného jazyka, ak sa na tom dohodnú v rámci protokolu a ani jeden nezmení svoj interface k vyššej a nižšej vrstve. Pritom o tejto zmene nikto iný ani nemusí vedieť. Podobne sekretárky môžu použiť e-mail, telefón bez toho, aby o tejto zmene čo len informovali vyššie vrstvy.

Každý proces môže pridať nejakú informáciu k dátam, ktoré posiela ďalej len pre svojho kolegu (peer) na druhej strane. Takáto informácia sa neposiela do vyššej vrstvy.

Viac technický pohľad ako môže vyzerať komunikácia medzi vrchnými vrstvami v 5-vrstvovej sieti.

- správu M vyprodukuje aplikácia bežiaca na 5. vrstve, pošle ju 4. vrstve, aby ju preniesla
- 4. vrstva pridá hlavičku H_4 (identifikáciu správy, napr. poradové číslo) a pošle to 3.vrstve. Hlavička umožní 4. vrstve na druhej strane doručovať správy pre vyššiu 5. vrstvu v správnom poradí za predpokladu, že nižšie vrstvy nezaručujú poradie správ
- obyčajne protokol 3. vrstvy nepodporuje správy neobmedzenej dĺžky, takže 3. vrstva musí rozdeliť prichádzajúcu správu od 4. vrstvy (H_4M) na menšie jednotky (packety H_4M_1 a M_2), pridať k nim hlavičku H_3, príp. vybrať výstupnú linku, cez ktorú to odošle 2. vrstve
- 2. vrstva pridá hlavičku H_2 aj chvost T_2 a výslednú jednotku pošle 1. vrstve na fyzický prenos
- na prijímajúcom počítači sa správa pohybuje smerom hore, každá vrstva spracuje a odoberie pre ňu určenú hlavičku (resp. chvost) a posunie správy vyššej vrstve


Layer
5
4
3
2
1
Layer 5 protocol
Layer 4 protocol
Layer 3 protocol
Layer 2 protocol
M
M
H4 M
H4 M
H3 H4 M1
H3 M2
H3 H4 M1
H3 M2
H2 H3 H4 M1 T2
H2 H3 M2 T2
H2 H3 H4 M1 T2
H2 H3 M2 T2
Source machine
Destination machine

Poznámka:

! vzťah medzi virtuálnou a skutočnou komunikáciou

! rozdiel medzi protokolom a interfaceom

Napr. peer procesy na 4. vrstve uvažujú o ich vzájomnej komunikácii ako o horizontálnej využívajúc protokol 4. vrstvy Každý z nich používa niečo ako procedúru ***SendToOtherSide*** resp. ***GetFromOtherSide***. Skutočne však obe procedúry komunikujú s nižšími vrstvami cez ***3/4 interface***.

Abstrakcia *peer* procesu bola kritickou v návrhu sietí. Obrovský nezvládnuteľný komplexný návrh siete bol rozdelený na menšie návrhové problémy, na návrh individuálnych vrstiev.

Poznámka:

Spodné vrstvy v hierarchii protokolov sa v praxi často implementujú ako hardware alebo firmware.

VŠEOBECNÉ ZÁVERY Z OBLASTI POČÍTAČOVÝCH SIETÍ, KTORÉ MUSIA BYŤ ZAKOMPONOVANÉ DO VIACVRSTVOVEJ SIEŤOVEJ ARCHITEKTÚRY

- mechanizmus na identifikáciu odosielateľov a príjemcov - **addressing**
- pravidlá pre prenos údajov - **simplex, half-duplex, full duplex**
  mnohé siete ponúkajú aspoň dva logické kanály pre jedno

  spojenie, jeden pre bežné a druhý pre naliehavé (urgent) údaje
- **správa chýb** - chyby detekujúce a chyby opravujúce kódy - fyzické komunikačné linky nie sú spoľahlivé. Mnoho kódov je známych, no obe strany komunikácie sa musia navzájom dohodnúť, ktorý použijú.prijímateľ musí mať možnosť povedať odosielateľovi, ktorá správa bola korektne prijatá a ktorá nie.

- **postupnosť (následnosť) správ**

  protokol musí dávať možnosť prijímateľovi znovu zoradiť kúsky správy dohromady najčastejšie sa používa očíslovanie, ale aj tak zostáva problém čo s tými, ktoré prišli mimo poradia

- **problém rýchleho odosielateľa a pomalého príjemcu**

najčastejšie riešenie uvažuje o nejakej spätnej väzbe od prijímateľa k odosielateľovi, èi už priamej alebo nepriamej,

o súèasnej situácii prijímateľa.

Iným riešením je limitovať odosielateľa na vopred dohodnutú prenosovú rýchlosť

- **neschopnosť akceptovať správy ľubovoľnej dĺžky**

  problém vedie k mechanizmu rozloženia, prenosu a opätovného zloženia správ

  disassembling, transmitting, reassembling

- **efektívny prenos malých správ**

  proces trvá na tom, aby sa prenášali dáta v tak malých jednotkách, že ich individuálne posielanie je neefektívne. Riešením je zhromaždiť niekoľko malých správ smerujúcich

  do spoločného cieľového uzla do jednej väčšej správy a jej rozčleneniu na druhej strane
- **multiplexing a demultiplexing**

  v situáciach, keď je nevhodné alebo drahé vytvoriť separované spojenie pre každú dvojicu komunikujúcich procesov, tak spodná vrstva môže využiť to isté spojenie pre niekoľko navzájom nesúvisiacich konverzácii

- **routing**

  existencia viacnásobných ciest medzi odosielateľom a príjemcom vedie k výberu a určeniu trasy

INTERFACES AND SERVICES

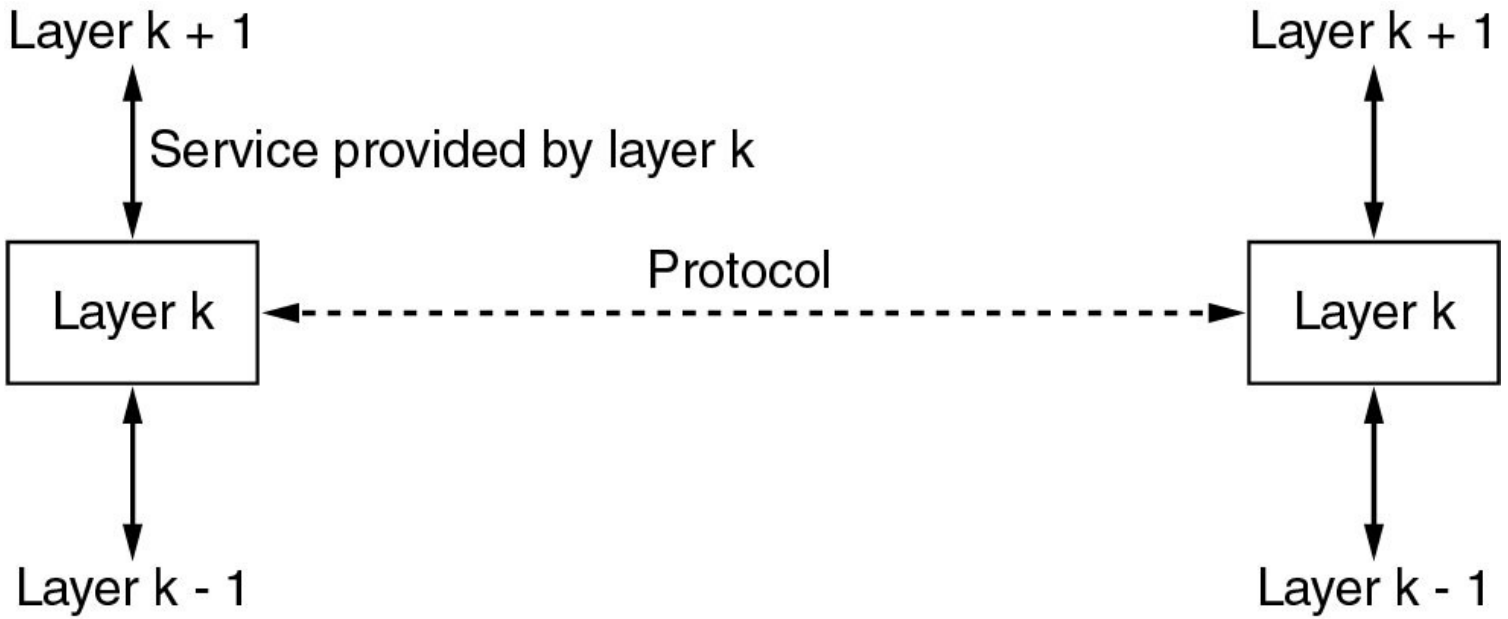
- **Entity - aktívne elementy v každej vrstve**

  softwarové (proces)

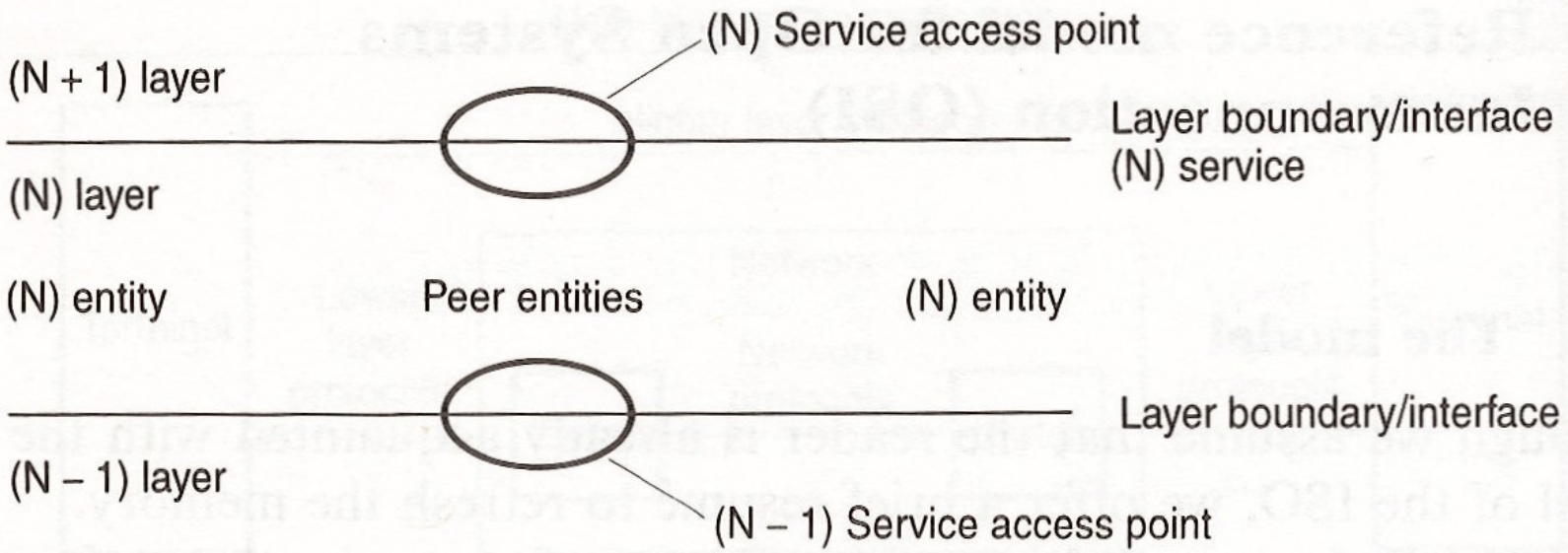
  hardwarové (inteligentný I/O chip)

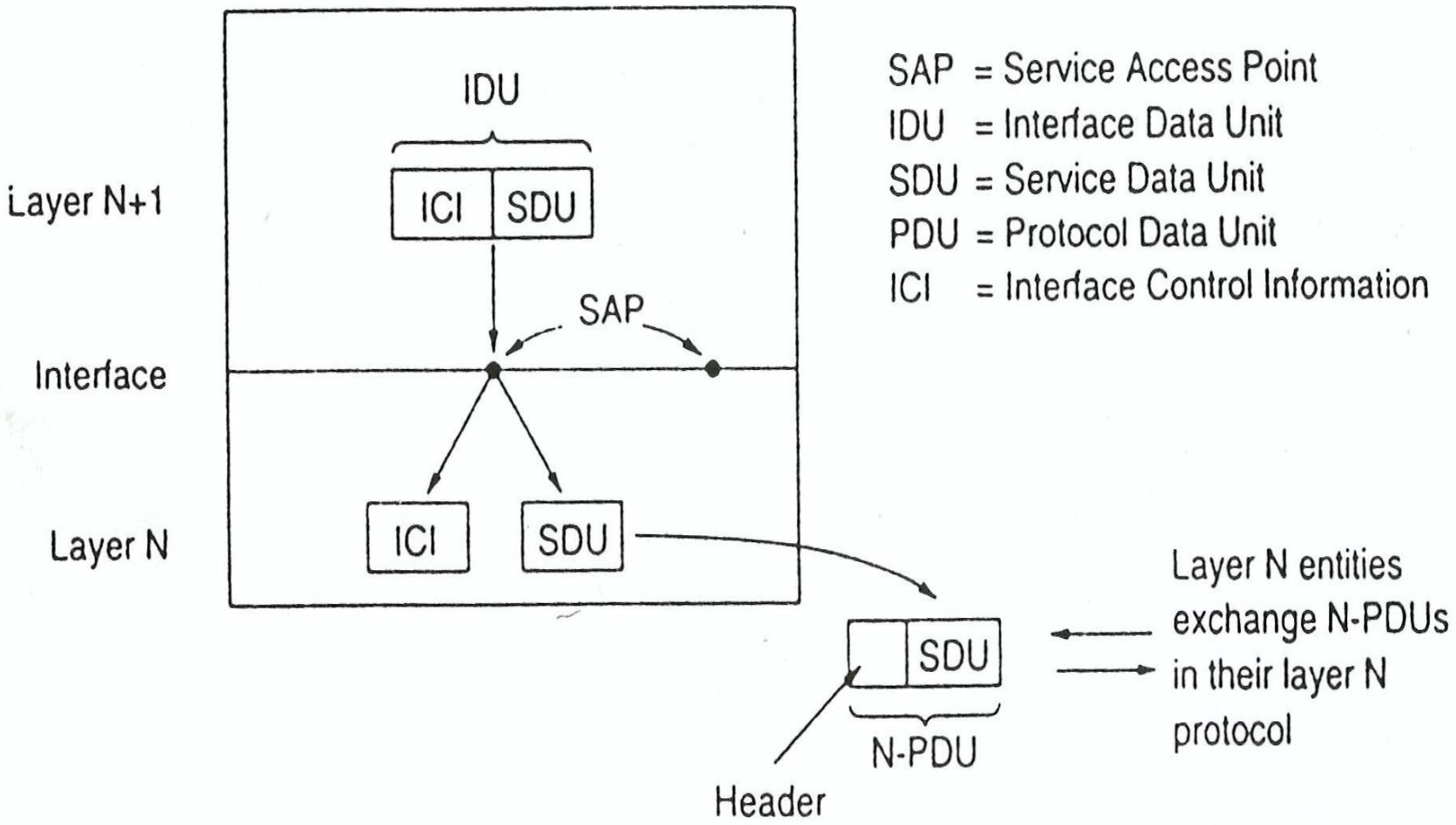
- **Peer entities - entity v tej istej vrstve na rôznych uzloch siete (napr. počítačoch, systémoch).**
- **Entity v N-tej vrstve implementujú služby využívané (N+1)-vrstvou. To znamená, že N-tá vrstva je poskytovateľ služby (service provider) a (N+1)-vrstva užívateľ danej služby (service user).**
- **N-tá vrstva môže využívať služby (N-1)-vrstvy za účelom poskytnutia služby. Môže ponúkať niekoľko tried služieb (rýchlu a drahú komunikáciu, resp. pomalú a lacnú komunikáciu)**

VZŤAH MEDZI VRSTVAMI A INTERFACE



PRINCÍPY VRSTVOVEJ ARCHITEKTÚRY - SAP




IDU
ICI
SDU
Layer N+1
SAP
Interface
Layer N
ICI
SDU
SDU
N-PDU
Header
SAP = Service Access Point
IDU = Interface Data Unit
SDU = Service Data Unit
PDU = Protocol Data Unit
ICI = Interface Control Information
Layer N entities
exchange N-PDUs
in their layer N
protocol

Service Access Points (SAPs) - miesta, kde sú jednotlivé služby k dispozícii. SAPs pre N-tú vrstvu sú miesta, kde môže N+1-vrstva pristupovať k jej ponúkaným službám.

Každý SAP jednoznačne identifikuje jeho adresa.

Príklad:

- **telefónny systém - SAPs sú telefónne zásuvky, do ktorých je možné pripojiť telefónny aparát a ich adresy sú telefónne čísla priradené týmto zásuvkám**
- **poštový systém - SAPs sú poštové schránky a im pride-lené adresy (meno, ulica, mesto, …)**

Aby došlo k výmene informácii medzi dvoma vrstvami musia existovať nejaké dohodnuté pravidlá týkajúce sa interface. Pri typickom interface entita z N+1-vrstvy pošle dátovú jednotku pre daný interface (Interface Data Unit -IDU) entite z N-tej vrstvy prostredníctvom SAP.

IDU pozostáva z dvoch častí

- **Service Data Unit (SDU)**
- **Interface Control Information (ICI)**

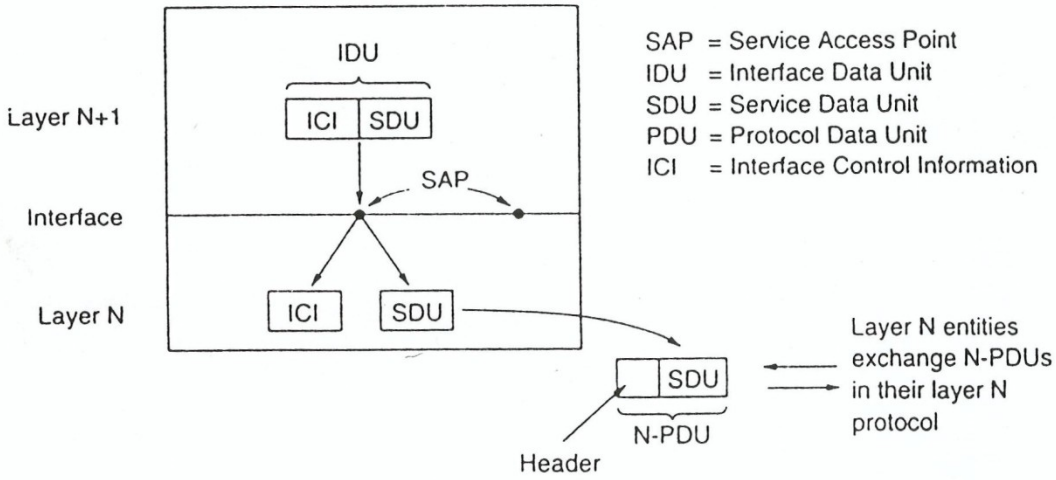
SDU - informácia, ktorá je poslaná prostrednictvom siete a určená pre entitu na rovnakej vrstve (peer entity) a odtiaľ smerom nahor do N+1-vrstvy.

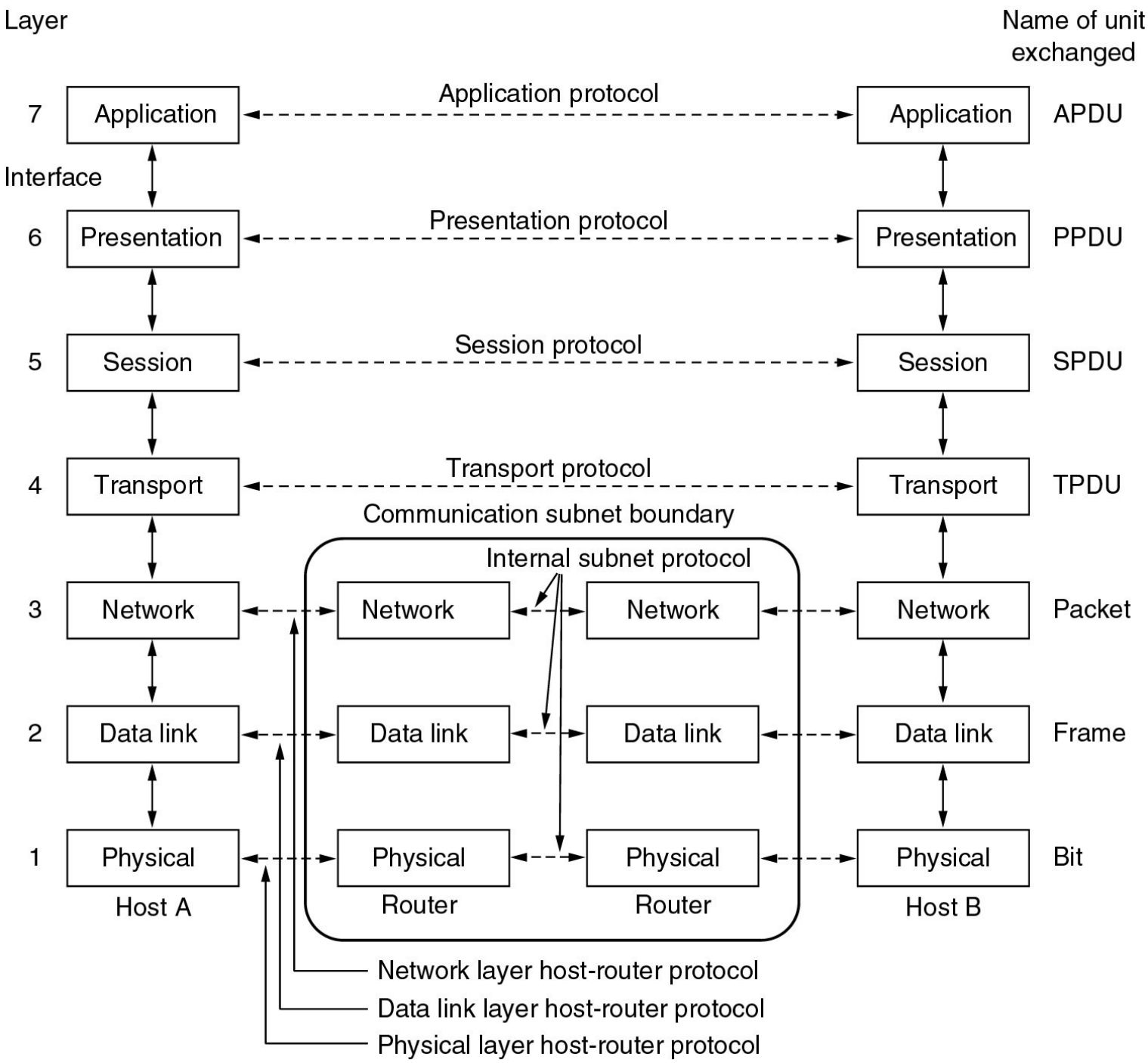
ICI - informácia, ktorá pomáha nižšej vrstve vykonať svoju prácu. Môže to byť napr. počet bytov v SDU. Nie je súčasťou prenášaných dát.

Aký môže byť ďalší osud SDU v N-tej vrstve?

Napríklad entita v N-tej vrstve musí urobiť jej fragmentáciu na menšie časti, každej z nich pridať hlavičku a poslať ich ako samostatné PDU do nižšej vrstvy.

Hlavičky sú určené ako pomoc pre entitu na rovnakej vrstve, aby mohol fungovať peer protocol N-tej vrstvy.




Layer
Name of unit exchanged
7
Application
Application protocol
Application
APDU
Interface
6
Presentation
Presentation protocol
Presentation
PPDU
5
Session
Session protocol
Session
SPDU
4
Transport
Transport protocol
Transport
TPDU
Communication subnet boundary
Internal subnet protocol
3
Network
Network
Network
Network
Packet
2
Data link
Data link
Data link
Data link
Frame
1
Physical
Physical
Physical
Physical
Bit
Host A
Router
Router
Host B
Network layer host-router protocol
Data link layer host-router protocol
Physical layer host-router protocol

SERVICES (SLUŽBY)

Jednotlivé sieťové vrstvy môžu poskytovať služby dvoch základných typov:

- CONNECTION - ORIENTED (SPOJOVANÉ)
- CONNECTIONLESS (NESPOJOVANÉ)

CONNECTION - ORIENTED
(SPOJOVANÉ)

Tieto služby môžu byť popísané na príklade telefónneho systému:

- zdvihni slúchadlo
- vytoč číslo
- rozprávaj
- polož slúchadlo

- pick up
- dial
- talk
- hang up

Podobne je to aj v prípade, keď užívateľ (service user) chce využiť na spojenie - orientovanú službu.

- vytvorenie spojenia
- využívanie spojenia
- uvoľnenie spojenia

- establish a connection
- use a connection
- release a connection

Takéto spojenie funguje ako „trubica“. Odosielateľ posiela bity na jednom konci, prijímateľ ich dostáva v nezmenenom poradí na druhom konci.

CONNECTIONLESS SERVICES (NESPOJOVANÉ)

Tieto služby môžu byť popísané na príklade poštového systému:

- Každá správa (list) si nesie so sebou úplnú cieľovú adresu.
- Každá správa je posielaná (routovaná) cez systém nezávisle od všetkých ostatných správ.
- Obyčajne, keď sa pošlú dve správy na tú istú cieľovú adresu, tak správa poslaná ako prvá príde skôr. Avšak môže sa stať, že prv poslaná správa sa niekde zdrží, takže ako prvá príde správa poslaná neskôr. Táto skutočnosť nie je možná pri spojovanej službe.

Služby všetkých vrstiev od aplikačnej po fyzickú môžu byť spojované (connection-oriented) alebo nespojované (connectionless). Na úrovni fyzickej vrstvy stráca zmysel rozlišovať medzi uvedenými typmi služieb.

Príklad:

na úrovni sieťovej vrstvy

vo verejných dátových sieťach sa nachádzajú služby, ktoré majú charakter:

spojovanej služby - prepájanie okruhov

nespojovanej služby - datagramová služba

na úrovni transportnej vrstvy

existuje nespojovaná služba, ktorá je implementovaná prostredníctvom služby sieťovej vrstvy spojovaného charakteru

KVALITA SLUŽBY

Niektoré systémy sú spoľahlivé v takom zmysle, že nikdy nestratia údaje. Obyčajne spoľahlivá služba je realizovaná prostredníctvom vhodného mechanizmu potvrdzovania príjmu (the receiver acknowledge the receipt of each message), takže odosielateľ si je istý, že bola doručená alebo príjemca žiada o znovuvyslanie chybných údajov. Potvrdzovací proces prináša určitú réžiu a zdržania, čo môže byť niekedy nežiadúce. Napr. pri prenose digitalizovaného zvuku je výhodnejšie občas prijať skreslený zvuk ako pripustiť výpadky.

Typickou situáciou, kedy spoľahlivá spojovaná služba je vhodná, je prenos súborov. Veľmi málo zákazníkov by si vybralo službu, ktorá občas premieša alebo stratí zopár bitov, aj keď by bola omnoho rýchlejšia.

Spoľahlivá spojovaná služba má dva varianty:

- **postupnosti správ (message sequences)**
  - hranice správ sú zachované

- **toky bytov (byte streams)**
  - terminal logging into remote system

Nie všetky aplikácie vyžadujú spojenie.

Napr. electronic junk mail - mať možnosť poslať jednoduchú správu s vysokou pravdepodobnosťou príchodu, ale bez garancie.

Nespoľahlivú (t.j. nepotvrdzovanú) nespojovanú službu nazývame datagram service. Podobne pri telegramoch sa nepotvrdzoval príjem späť k odosielateľovi.

Niekedy sa uvažuje o vymoženosti poslať krátku správu bez vytvorenia spojenia, ale spoľahlivosť sa vyžaduje.

V takých situáciach sa použije tzv.
potvrdzovaná datagramová služba
(the acknowledged datagram service).

Pripomína to situáciu, keď pošlete registračný formulár bežnou poštou a žiadate potvrdenie o registrácii. Ak dostanete potvrdenie späť, tak viete naisto, že váš formulár bol doručený a že sa cestou nestratil.

Request-reply service (odpoveď na požiadavku)

- odosielateľ pošle jednoduchý datagram obsahujúci požiadavku

- odozva obsahuje odpoveď na požiadavku.

Služba sa používa na implementáciu komunikácie v mo-deloch klient-server, klient pošle požiadavku, server na ňu reaguje.

	SERVICE	EXAMPLE
Connection--oriented	Reliable message stream	Sequence of pages
	Reliable byte stream	Remote login
	Unreliable connection	Digitized voice
Connectionless	Unreliable datagram	Electronic junk mail
	Acknowledged datagram	Registered mail
	Request-reply	Database query

SLUŽBOVÉ PRIMITÍVA (SERVICE PRIMITIVES)

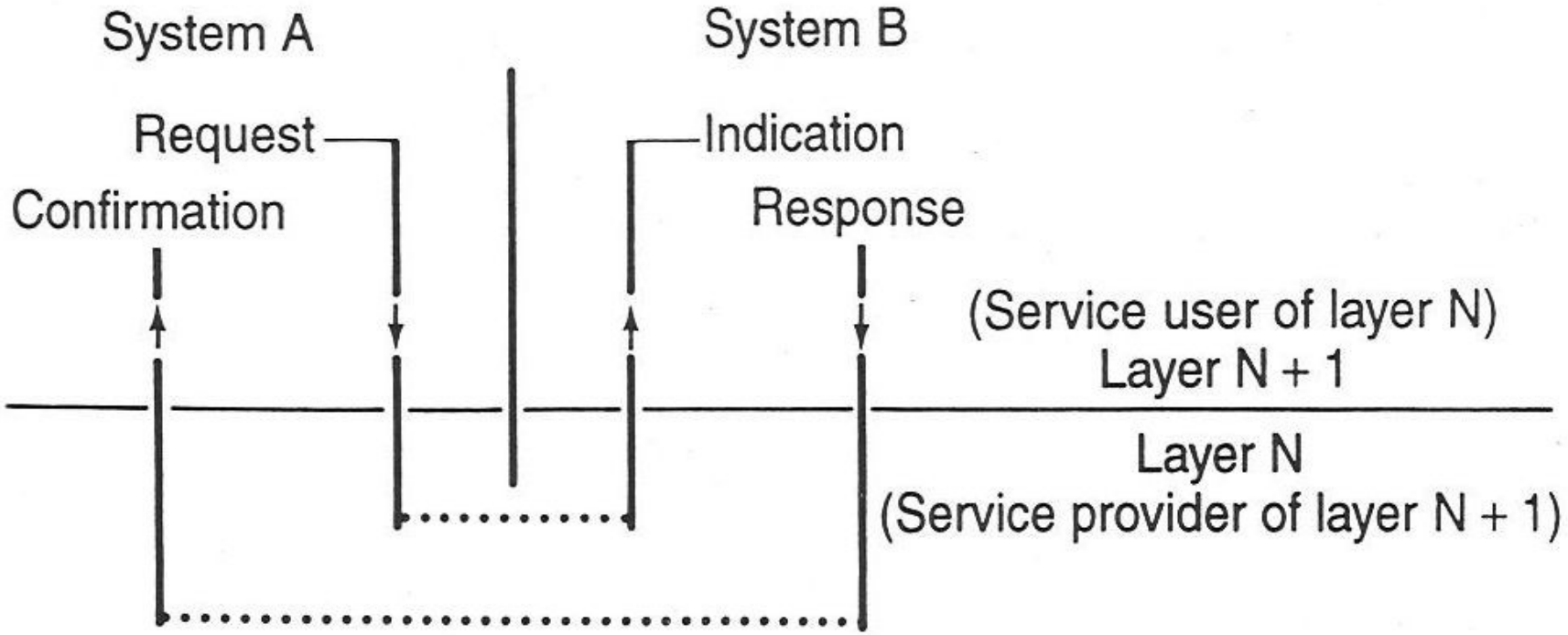
Mechanizmus využívania služieb na rozhraní vrstiev je formálne špecifikovaný skupinou základných operácií - tzv. primitívami, z ktorých služby pozostávajú.

Primitíva povedia službe, aby vykonala nejakú akciu alebo aby oboznámila o akcii vykonanej partnerskou entitou. Môžme ich rozdeliť do štyroch základných tried

Základné triedy službových primitív

Primitive	Meaning (Význam)
Request (Žiadosť)	Entita žiada službu o vykonanie nejakej činnosti
Indication (Indikácia)	Entita má byť informovaná o určitej činnosti
Response (Odozva)	Entita chce reagovať na určitú činnosť
Confirm (Potvrdenie)	Entita je informovaná o výsledku vybavenia svojej požiadavky

SLUŽBOVÉ PRIMITÍVA



Ilustrujme použitie primitív uvažujúc o tom ako je spojenie vytvorené (established) a uvoľnené (released).

Entita inicializujúca spojenie urobí
CONNECT-REQUEST
dôsledkom čoho je poslanie packetu.

Prijímateľ dostáva CONNECT-INDICATION
že nejaká entita chce vytvoriť spojenie s ním.

Entita, ktorá dostala connect-indication, použije
CONNECT-RESPONSE,
aby oznámila, či návrh prijíma alebo zamieta.

Tak či onak, entita, ktorá inicializovala connect-request, zistí čo sa stalo prostredníctvom
CONNECT-CONFIRM.

Primitíva môžu obsahovať parametre.

connect-request - meno volaného uzla, typ požadovanej služby, dĺžku prenášanej správy

connect-indication - identifikáciu volajúceho, typ požadovanej služby, navrhovanú maximálnu dĺžku správy

connect-response - ak volaná entita nesúhlasí s navrhovanou maximálnou dĺžkou správy, urobí protinávrh

connect-confirm - protinávrh by mal byť k dispozícii pôvodnému volajúcemu

Detaily (pravidlá) takéhoto vyjednávania sú obsiahnuté v protokole príslušnej vrstvy. Napr. v protokole môže byť určené, že v prípade nezhody, maximálnou dĺžkou správy bude menšia z navrhovaných.

V potvrdzovanej službe (confirmed service) existuje

Request (Žiadosť)
Indication (Indikácia)
Response (Odozva)
Confirm (Potvrdenie)**.**

V nepotvrdzovanej službe (unconfirmed service) existuje len

Request (Žiadosť)
Indication (Indikácia).

CONNECT

- je vždy potvrdzovaná služba, lebo vzdialený partner (remote peer) musí súhlasiť s vytvorením spojenia.

DATA TRANSFER

- môže byť potvrdzovaná aj nepotvrdzovaná služba, v závislosti na tom, či odosielateľ požaduje potvrdenie príjmu.

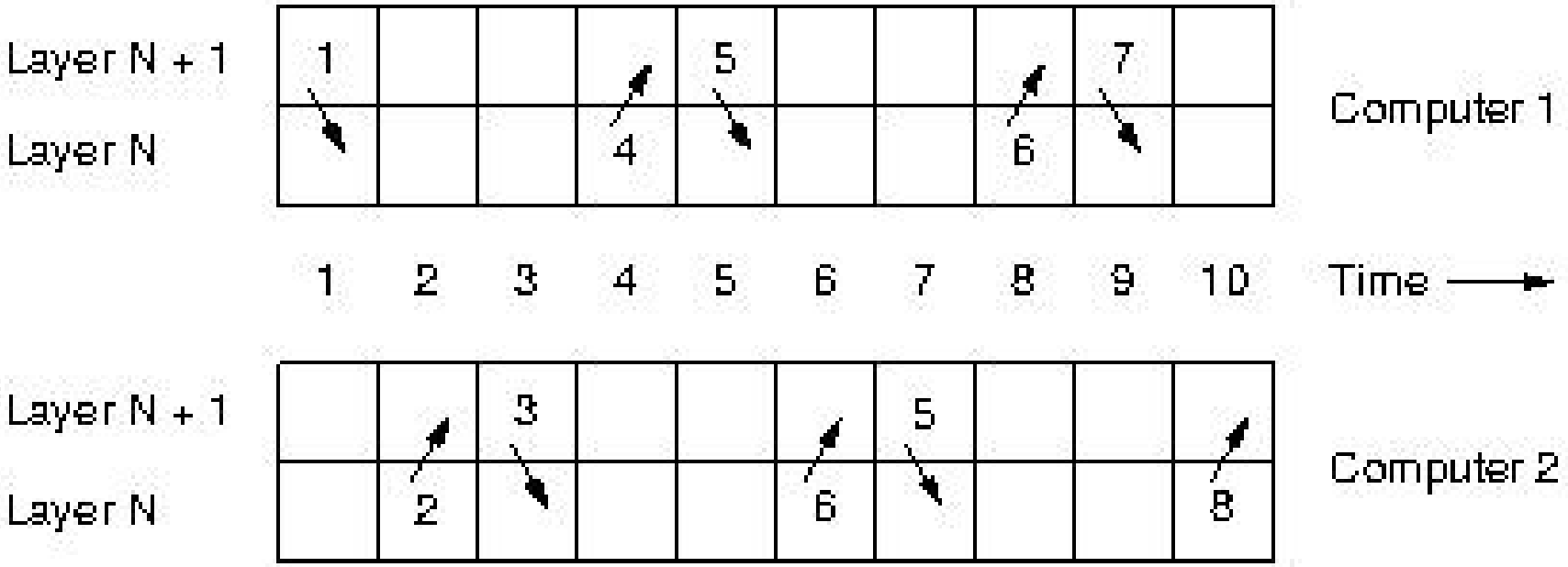
Príklad spojovanej služby obsahujúcej 8 službových primitív:

1. **connect request - žiadosť na vytvorenie spojenia**
2. **connect indication - signalizovanie volanej strane**
3. **connect response - využíva volaný na akceptovanie/zamietnutie volaní**
4. **connect confirm - oznámenie volajúcemu, či volanie bolo akceptované**
5. **data request - požiadavka, že údaje budú poslané**
6. **data indication - signalizovanie prichádzajúcich údajov**
7. **disconnect request - žiadosť o uvoľnenie spojenia**
8. **disconnect indication - signalizovanie volanej strane, že je požiadavka na zrušenie spojenia**

V tomto príklade platí:
CONNECT je potvrdzovaná služba (explicitná odozva sa požaduje).
DISCONNECT je nepotvrdzovaná služba (žiadna odozva).

Analógia s telefónnym systémom je dobrým príkladom na vysvetlenie funkcie primitív:

1. connect request - vytočenie čísla volaného
2. connect indication - telefón volaného zvoní
3. connect response - volaný zodvihne slúchadlo
4. connect confirm - volajúci počuje, že skončilo vyzváňanie
5. data request - volajúci pozve volaného na kávu
6. data indication - volaný počuje pozvanie
7. data request - volaný natešene súhlasí
8. data indication - volajúci počuje natešený súhlas
9. disconnect request - volajúci zavesí slúchadlo
10. disconnect indication - volaný počuje zavesenie slúchadla a zavesí tiež



Čísla pri koncoch šípok odkazujú na osem službových primitív diskutovaných vyššie.

VZŤAH SLUŽIEB A PROTOKOLOV

Služby a protokoly sú dva rôzne koncepty.

Služba - skupina primitív (operácií), ktoré vrstva poskytuje vrstve nad ňou
- **definuje, ktoré operácie je vrstva pripravená poskytnúť pre svojich užívateľov**
- **nehovorí nič o tom ako sú operácie implementované**
- **má vzťah k interface medzi dvoma vrstvami**
  - **service user - vyššia vrstva**
  - **service provider - nižšia vrstva**

Protokol - množina pravidiel riešiacich formát a význam rámcov, packetov, správ, ktoré sa vymieňajú medzi entitami rovnakej vrstvy. Entity využívajú protokoly, aby mohli implementovať definície svojich služieb. Majú voľnosť v tom, že môžu zmeniť protokol danej vrstvy, pokiaľ nezmenia službu viditeľnú pre ich užívateľov.

Protokol a služba sú úplne oddelené.

Služba je ako abstraktný dátový typ alebo objekt v objektovo-orientovanom programovacom jazyku. Dátový typ definuje operácie, ktoré môžu byť s objektom vykonané, ale nie je špecifikované ako sú tieto operácie implementované.

Protokol sa vzťahuje k implementácii služby a taktiež nie je viditeľný pre užívateľa služby.