Reti di Comunicazioni e Internet

Strato di trasporto (Generalità & UDP)

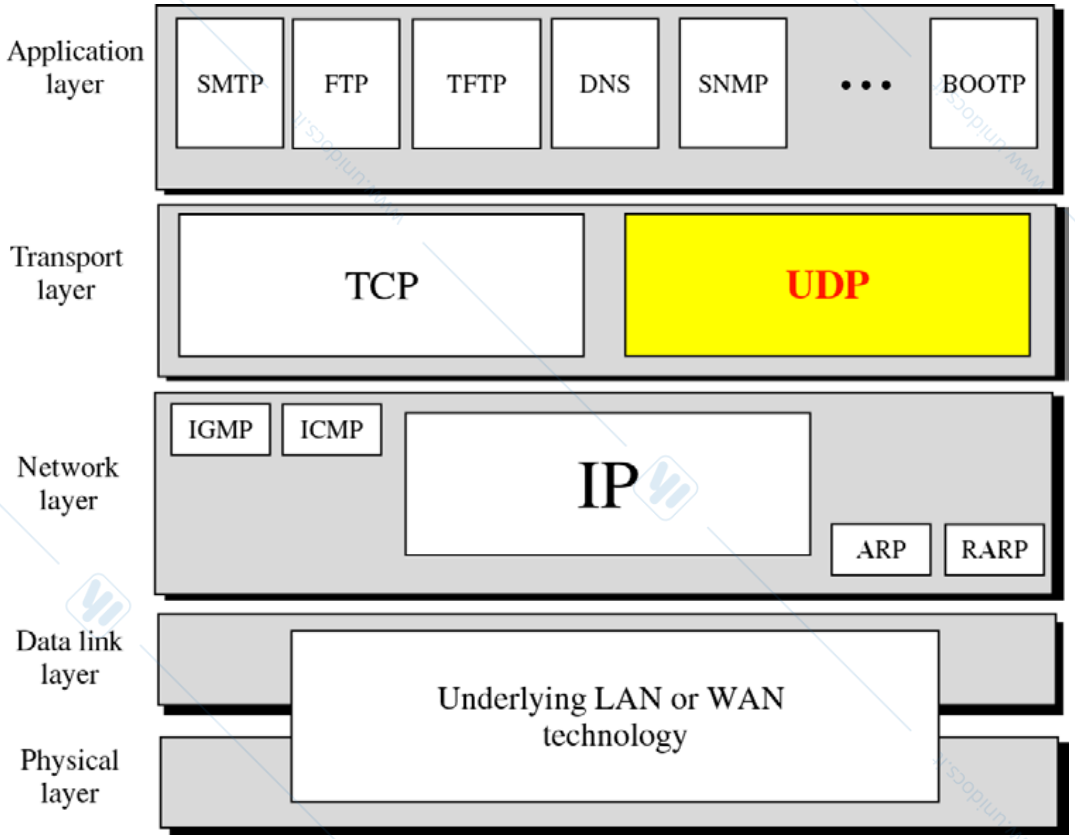
Massimo Tornatore
Politecnico di Milano

Dip. di Elettronica, Informazione e Bioingegneria (DEIB)

Via Ponzio, 34/5 – 20133 Milan, Italy
massimo.tornatore @polimi.it

Strato di Trasporto

- **Introduzione**
- **Protocollo UDP**
- **Protocollo TCP**



Outline

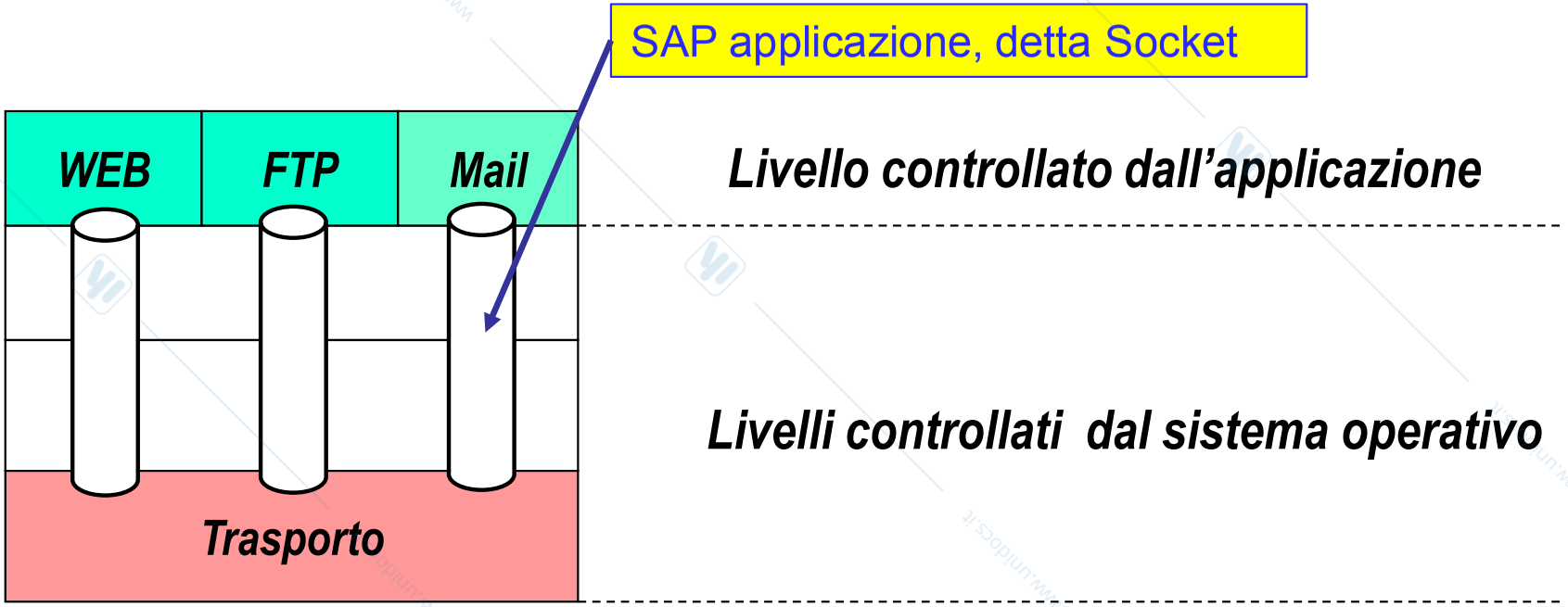
- **Introduzione**
  - Comunicazioni tra processi
  - Servizi del livello di trasporto
  - Comunicazioni client-server
- **Protocollo UDP**
- **Protocollo TCP**

Comunicazione tra processi

- **Processo: programma in esecuzione su un host**
- **Nello stesso host:**
  - Processi comunicano usando comunicazione inter-processo (definita dall'OS)
- **In host differenti:**
  - Processi comunicano scambiando messaggi

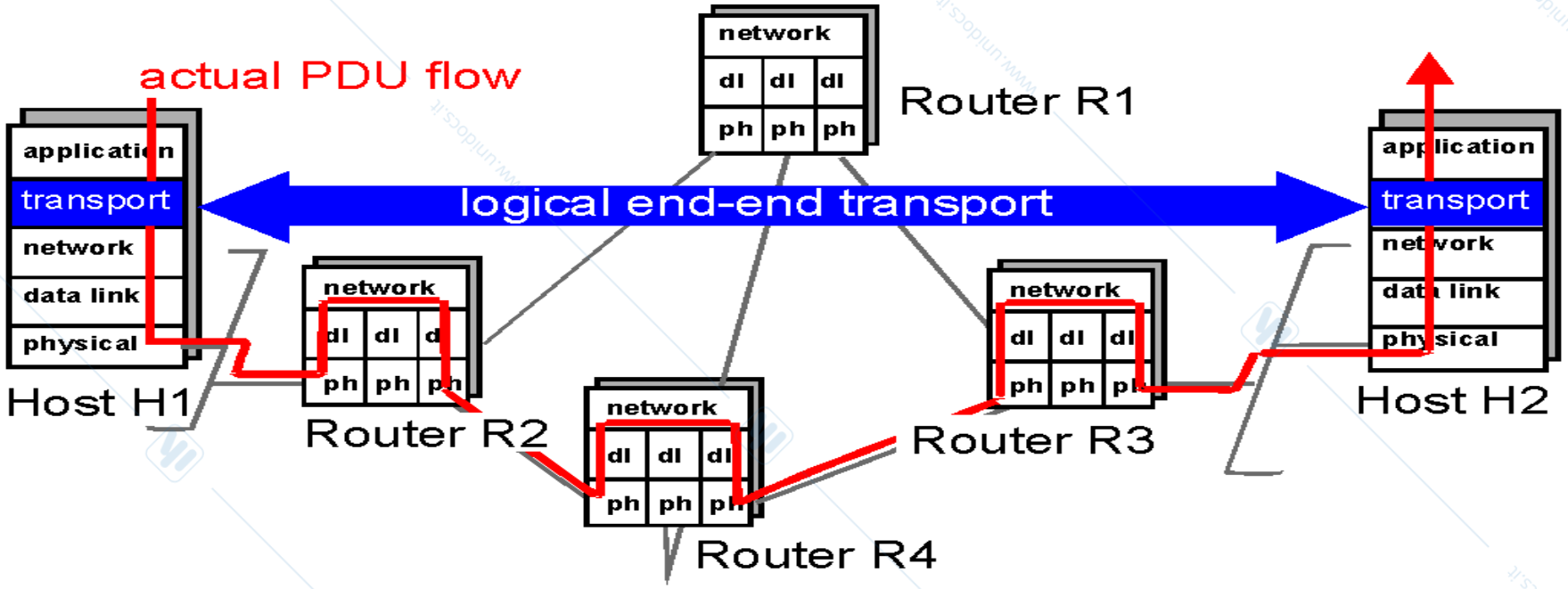
Interazione coi livelli inferiori

- Lo scambio di messaggi fra i protocolli applicativi avviene utilizzando i servizi dei livelli inferiori attraverso i SAP (*Service Access Point*)
- Nella architettura a strati di Internet (TCP/IP), i protocolli applicativi si appoggiano direttamente sul livello di trasporto



Servizio di trasporto

- Obiettivo: instaurare un collegamento logico tra applicazioni su host remoti
  - Dal punto di vista delle applicazioni, l'intera rete è vista come un collegamento *diretto* tra gli host su cui si eseguono i processi



Outline

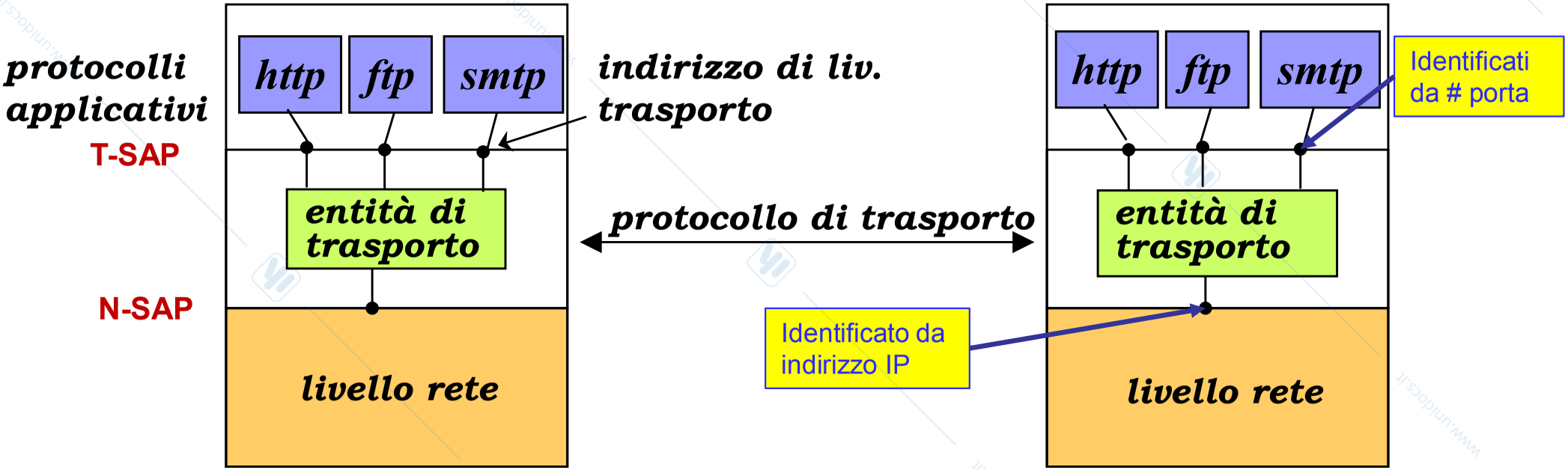
- **Introduzione**
  - Comunicazioni tra processi
  - Servizi del livello di trasporto
  - Comunicazioni client-server
- **Protocollo UDP**
- **Protocollo TCP**

Servizi del livello di trasporto

- **Multiplazione**
- **Indirizzamento**
- **Buffering**

Servizio di trasporto: funzione di *mux/demux*

- Più applicazioni possono essere attive su un *host*
  - il livello di trasporto svolge funzioni di ***multiplexing/demultiplexing***



Servizio di trasporto: funzione di *Indirizzamento*

- **Per poter indicare univocamente il processo destinatario è necessario un *identificativo***
- ***Domanda:* è sufficiente l'indirizzo IP a identificare un processo in esecuzione su una macchina?**
- ***Risposta:* NO, perché più di un processo può essere in esecuzione sullo stesso host contemporaneamente**
  - Quindi anche il livello di trasporto effettua **multiplazione** (di messaggi di più applicazioni sulla stessa rete)

Servizio di trasporto: funzione di *Indirizzamento*

- ***L'identificativo* include:**
  - *indirizzo IP*, associato all'host
  - *numero di porta*, associato ad un processo in esecuzione sull'host
- **Ad esempio per inviare un messaggio al server web www.polimi.it:**
  - IP address: 131.175.12.34
  - Port number: 80

Servizio di trasporto: funzione di *Indirizzamento*

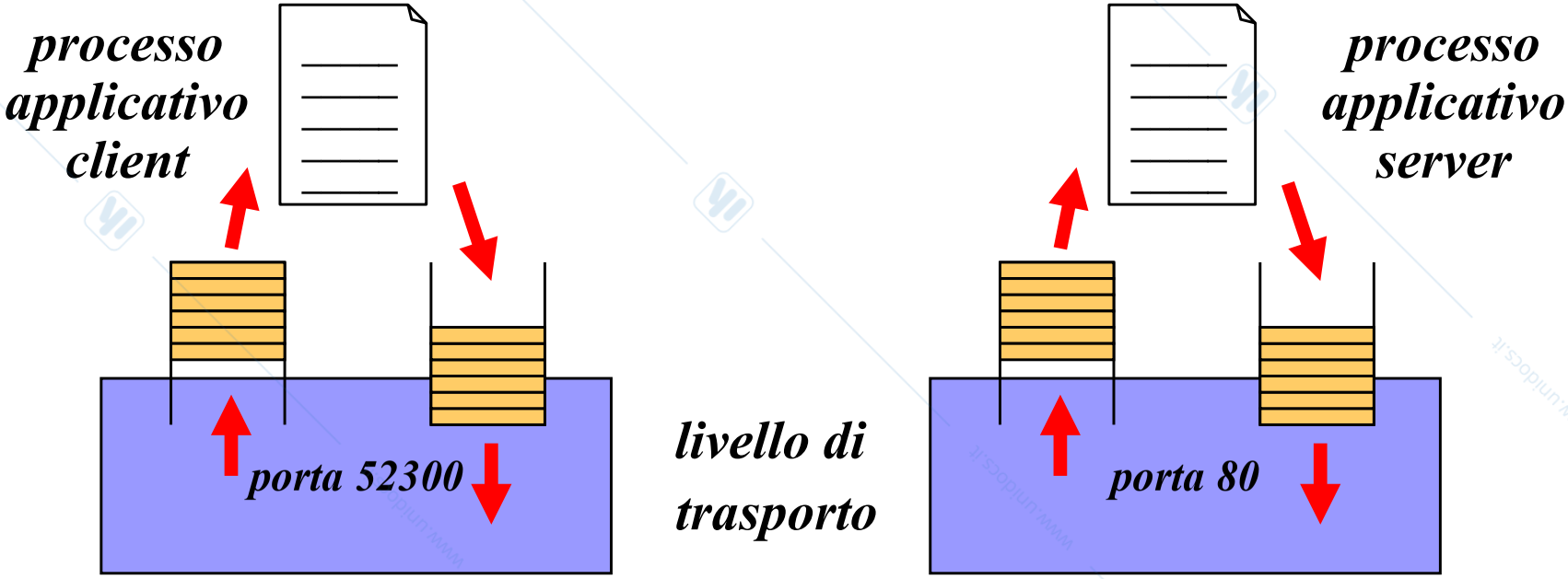
- Le funzioni di *multiplexing/demultiplexing* vengono gestite mediante indirizzi contenuti nelle UI di livello di trasporto (segmenti)
- Tali indirizzi sono lunghi *16 bit* e prendono il nome di ***porte***
- I numeri di porta possono assumere valori compresi tra 0 e 65535 (=2^{16}-1) e si dividono in 3 gruppi:
  - ***Well-known ports***: sono assegnati ad importanti applicativi dal lato *server* (HTTP, FTP, SMTP, DNS, ecc.)
  - ***Numeri registrati***: assegnati a specifiche applicazioni da chi ne faccia richiesta (tipicamente protocolli proprietari)
  - ***Numeri dinamici***: assegnati dinamicamente ai processi applicativi lato *client*

Well-known ports

- **Alcune porte «famose» assegnate su base universale («well-known ports)**
  - FTP (TCP): 21
  - Telnet (TCP): 23
  - SMTP (TCP): 25
  - DNS (TCP/UDP): 53
  - BOOTP server (UDP): 67
  - BOOTP client (UDP): 68
  - BOOTP (TCP): 69
  - Finger (TCP): 79
  - WWW-HTTP (TCP): 80

Servizio di trasporto: funzione di *buffering*

- Quando un processo viene associato ad una porta (lato *client* o lato *server*) viene associato dal sistema operativo a ***due code (buffer)***, una d'ingresso e una d'uscita
  - I buffer "assorbono" i rallentamenti del processing dei livelli adiacenti (rete e applicazione)



Servizio di trasporto: diverse modalità

- Il servizio di rete è non affidabile
  - Fa del proprio meglio per consegnare i singoli messaggi a destinazione ("best-effort")
- Si possono fornire vari tipi di servizio di trasporto
  - ***Affidabile*** (garanzia di consegna dei messaggi nel corretto ordine)
  - ***Non affidabile*** (solo funzionalità di multiplexing)
  - ***Connection-oriented***
  - ***Connectionless***
- In Internet sono definiti due tipi di trasporto
  - TCP (Transmission Control Protocol): connection-oriented e affidabile (sopperisce alle mancanze del liv. di rete)
  - UDP (User Datagram Protocol): connectionless e non affidabile

Applicazioni e protocolli di trasporto

- **Applicazioni diffuse in Internet e relativi protocolli di trasporto**

Applicazione	Protocollo dello strato dell'applicazione	Protocollo di trasporto adottato
Posta elettronica	SMTP	TCP
Accesso a terminale remoto	Telnet	TCP
Web	HTTP	TCP
Trasferimento di file	FTP	TCP
Server di file remoto	NFS	tipicamente UDP
Streaming multimediale	proprietario	tipicamente UDP
Telefonia Internet	proprietario	tipicamente UDP
Gestione della rete	SNMP	tipicamente UDP
Protocollo di routing	RIP	tipicamente UDP
Traduzione del nome	DNS	tipicamente UDP

Outline

- **Introduzione**
- **Protocollo UDP**
  - Pattavina 22.2, Kurose 3.3
- **Protocollo TCP**

Protocolli di trasporto
UDP

- **User Datagram Protocol (UDP): RFC 768**
  - Protocollo connectionless
  - Fornisce
    - **Multiplazione per mezzo dei socket**
    - **Rivelazione di errore opzionale**
  - Non fornisce
    - Consegna in sequenza
    - Rivelazione di perdita/duplicazione
    - Controllo di flusso

→Servizio di trasporto ***inaffidabile***

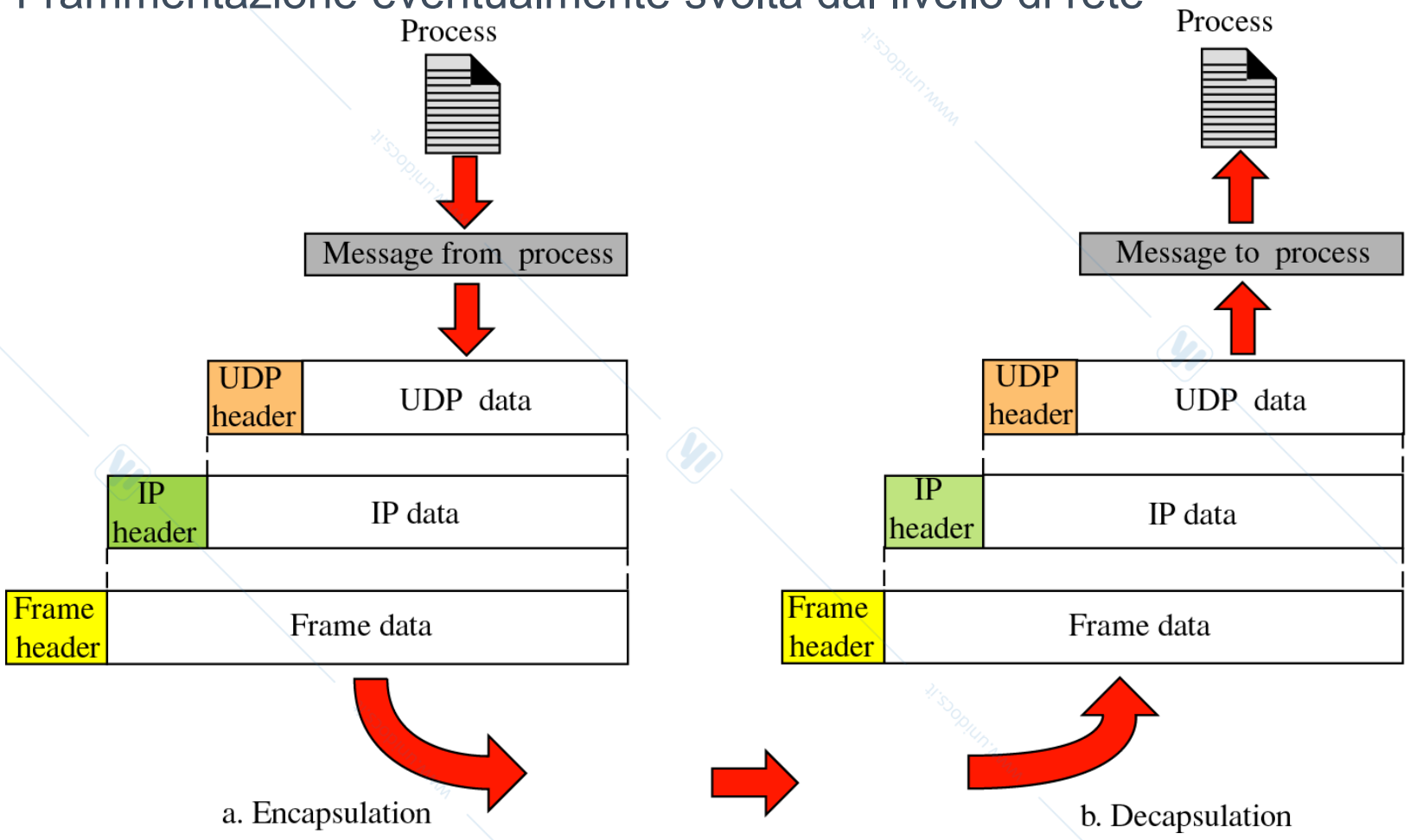
Non offre particolari servizi aggiuntivi rispetto al livello sottostante (IP)

Perchè usare UDP e non TCP?

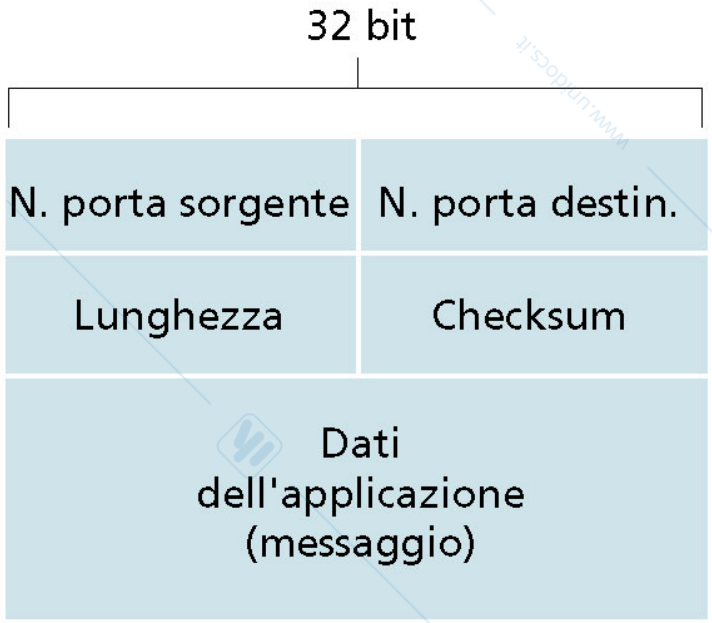
- **Minore latenza**
  - Non occorre stabilire una connessione
- **Maggiore semplicità**
  - Non occorre tenere traccia dello stato della connessione
  - Poche regole da implementare
- **Minore overhead**
  - Header UDP è più piccolo dell'header TCP

UDP
Incapsulamento

- **Messaggio applicativo incapsulato ESATTAMENTE in 1 messaggio UDP e trasportato quindi da 1 pacchetto IP**
  - Frammentazione eventualmente svolta dal livello di rete



UDP
Header



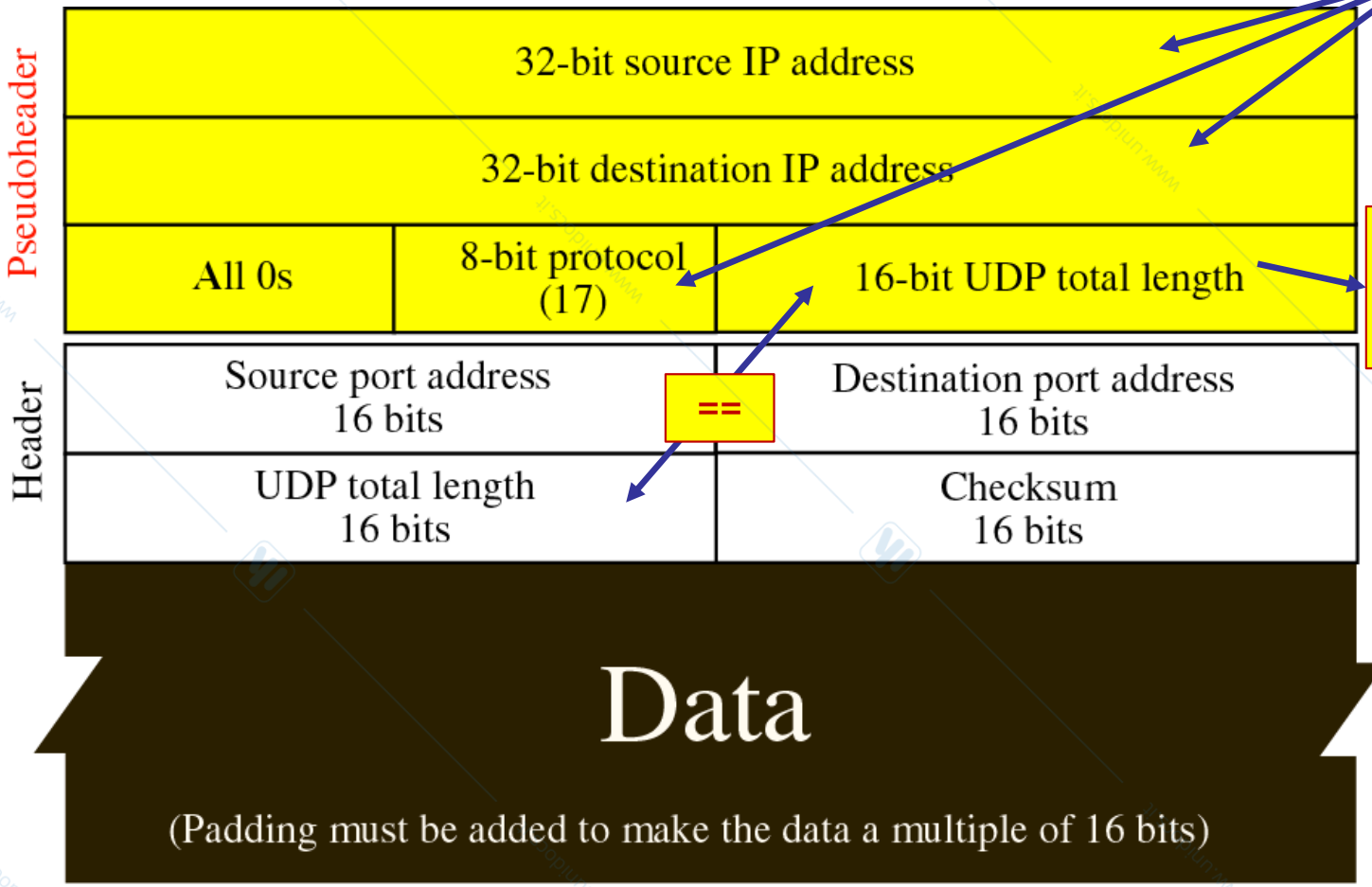
- **Porta sorgente**
- **Porta destinazione**
- **Lunghezza: lunghezza di header + data**
- **Checksum: è opzionale**
  - Protegge contro la consegna sbagliata da parte di IP
  - Copertura include UDP header + UDP data + pseudoheader (IP source, IP destination, UDP protocol number, byte count for UDP segment)

UDP

Calcolo del checksum e Pseudoheader

PSEUDOHEADER: definito solo ai fini del calcolo del checksum (per evitare recapiti di UDP a host sbagliati)

Campi copiati da header IP



Attenzione! (non comprende lo pseudo header né eventuale padding)

Protocol 17 = UDP

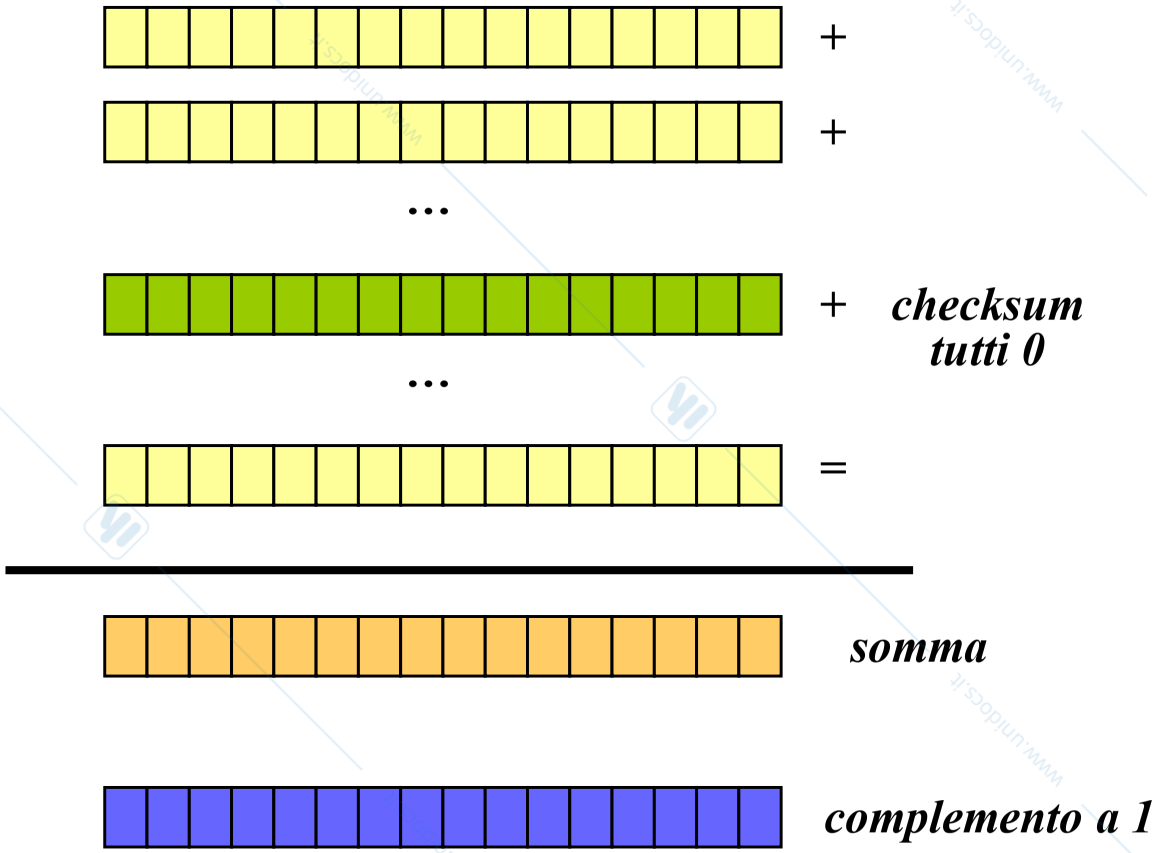
UDP
Checksum

- **Procedura di calcolo**
  - Segmento UDP + pseudoheader allineato a 16 bit (padding eventuale)
  - **Sum** = Somma in ARITMETICA COMPLEMENTO A 1 di tutte le righe
    - L'eventuale riporto viene aggiunto al risultato come bit meno significativo
  - **Checksum** = COMPLEMENTO A 1 (= NOT) della somma precedente
  - ⇒ In ricezione Sum+Checksum deve dare 0 in assenza di errori

	1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0
	1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
wraparound	1	1	0	1	1	1	0	1	1	1	0	1	1	1	0	1	1
sum	1	0	1	1	1	0	1	1	1	0	1	1	1	1	0	0
checksum	0	1	0	0	0	1	0	0	0	1	0	0	0	0	1	1

Calcolo del Checksum lato trasmettitore

- Segmento + pseudo-header sono divisi in blocchi da 16 bit
- Il *Checksum* è inizializzato a 0
- Tutti i blocchi vengono sommati complemento a 1
- Il risultato è complementato ed inserito nel campo di *checksum*



UDP
Esempio

- ***Trasmissione***

153.18.8.105			
171.2.14.10			
All 0s	17	15	
1087		13	
15		All 0s	
T	E	S	T
I	N	G	All 0s

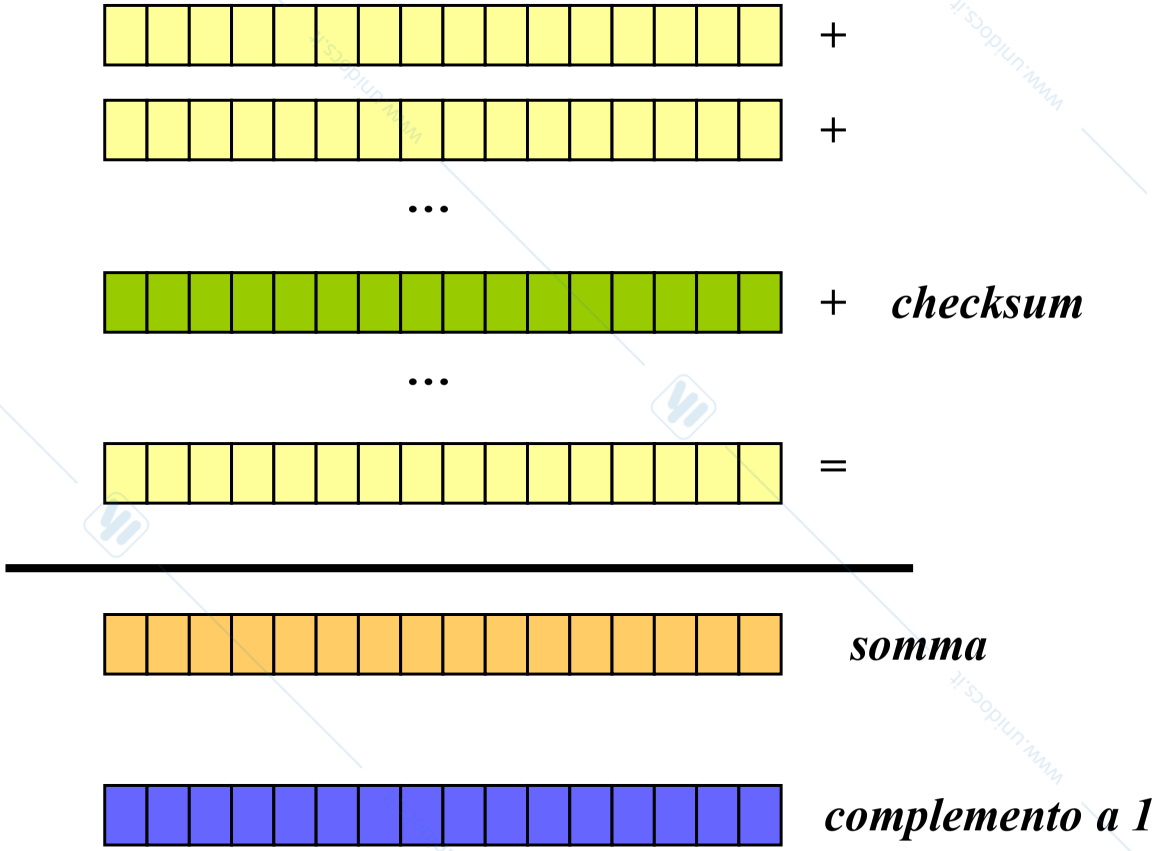
```
10011001 00010010 ──→ 153.18
00001000 01101001 ──→ 8.105
10101011 00000010 ──→ 171.2
00001110 00001010 ──→ 14.10
00000000 00010001 ──→ 0 and 17
00000000 00001111 ──→ 15
00000100 00111111 ──→ 1087
00000000 00001101 ──→ 13
00000000 00001111 ──→ 15
00000000 00000000 ──→ 0 (checksum)
01010100 01000101 ──→ T and E
01010011 01010100 ──→ S and T
01001001 01001110 ──→ I and N
01000111 00000000 ──→ G and 0 (padding)
-------------------------------------
10010110 11101011 ──→ Sum
01101001 00010100 ──→ Checksum   50928
```

La stringa (50928 in binario) viene inserita nel campo checksum prima di trasmettere il pacchetto

Il padding non viene inviato! Serve solo a fare i conti del checksum!

Calcolo del Checksum lato ricevitore

- **Segmento + pseudo-header sono divisi in blocchi da 16 bit**
- **Tutti i blocchi vengono sommati complemento a 1**
- **Il risultato è complementato**
  - Se sono tutti 0 il segmento è accettato
  - Altrimenti è scartato



UDP
Esempio

- ***Ricezione senza errori***

153.18.8.105		
171.2.14.10		
All 0s	17	15
1087		13
15		50928
T	E	S	T
I	N	G	All 0s

```
10011001 00010010 ──→ 153.18
00001000 01101001 ──→ 8.105
10101011 00000010 ──→ 171.2
00001110 00001010 ──→ 14.10
00000000 00010001 ──→ 0 and 17
00000000 00001111 ──→ 15
00000100 00111111 ──→ 1087
00000000 00001101 ──→ 13
00000000 00001111 ──→ 15
01101001 00010100 ──→ (checksum)
01010100 01000101 ──→ T and E
01010011 01010100 ──→ S and T
01001001 01001110 ──→ I and N
01000111 00000000 ──→ G and 0 (padding)
-----------------------------------
11111111 11111111 ──→ Sum
00000000 00000000 ──→ 1 – Sum
```

UDP
Esempio

- ***Ricezione <u>con errore</u>***

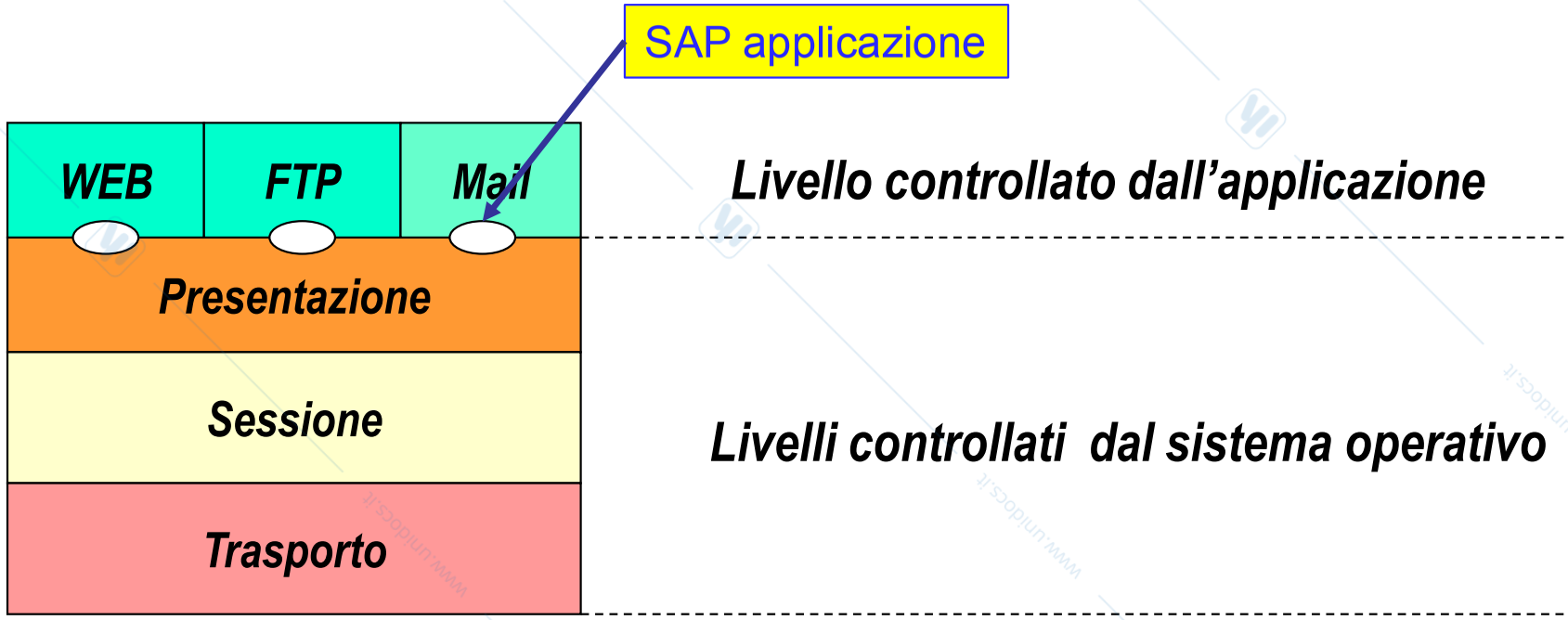
153.18.8.105		
171.2.14.10		
All 0s	17	15
1087		13
15		50928
T	E	S	T
I	N	G	All 0s

```
10011001 00010010 ──→ 153.18
00001000 01101001 ──→ 8.105
10101011 00000010 ──→ 171.2
00001110 00001010 ──→ 14.10
00000000 00010001 ──→ 0 and 17
00000000 00001111 ──→ 15
00000100 00111111 ──→ 1087
00000000 00001101 ──→ 13
00000000 00001111 ──→ 15
01101001 00010100 ──→ (checksum)
01010100 01000101 ──→ T and E
01010011 01010100 ──→ S and T
01001001 01001110 ──→ I and N
01000111 00000001 ──→ G and 0 (padding)
-----------------------------------
00000000 00000001 ──→ Sum
11111111 11111110 ──→ 1 – Sum
```

FINE

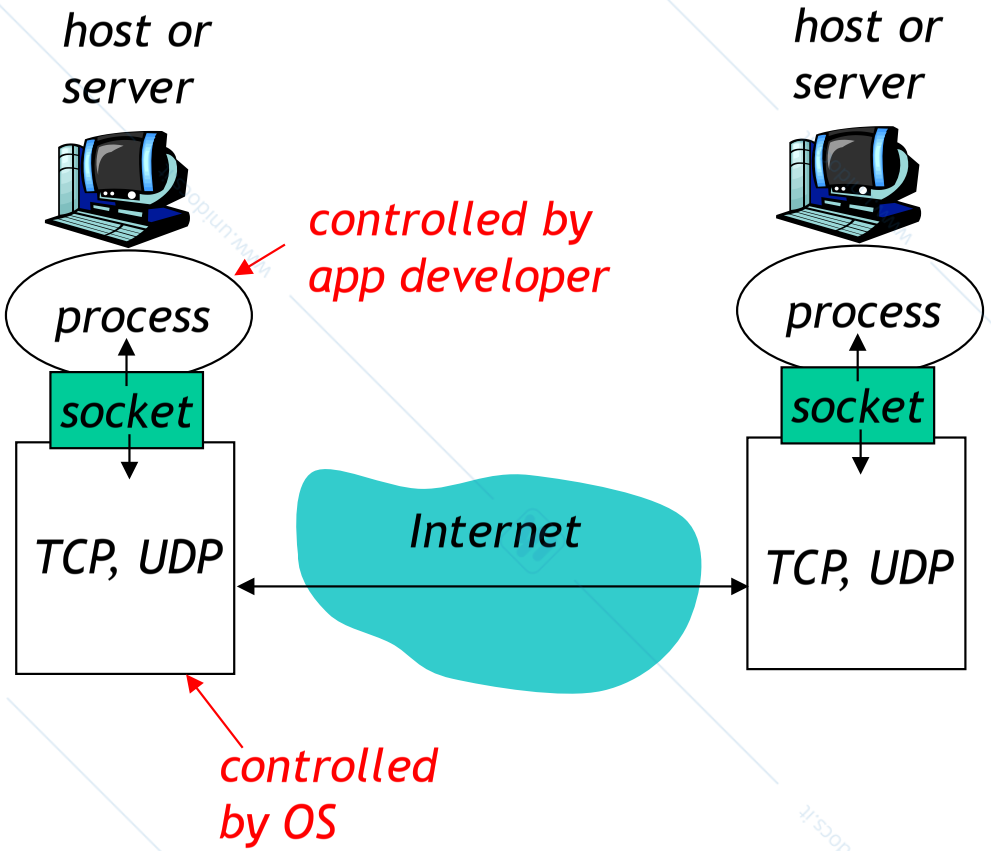
Interazione coi livelli inferiori

- **Lo scambio di messaggi fra i protocolli applicativi avviene utilizzando i servizi dei livelli inferiori attraverso i SAP (*Service Access Point*)**
- **Ogni processo è associato ad un SAP**
- **Nella pila OSI:**



Socket

- **Processi inviano e ricevono messaggi attraverso i socket**
- **I socket sono delle porte di comunicazione**
  - Il processo trasmittente scrive il messaggio sulla porta (socket)
  - La rete raccoglie il messaggio e lo trasporta fino alla porta del destinatario
  - Il processo destinatario riceve il messaggio attraverso la porta (socket) di destinazione



- *I socket sono i SAP tra il livello applicativo e il livello di trasporto*

Servizio di trasporto

- Il livello di trasporto ha il compito di instaurare un **collegamento logico** tra le applicazioni residenti su host remoti
- Il livello di trasporto rende trasparente il trasporto fisico dei messaggi alle applicazioni