


Linguaggi ed Applicazioni multimediali

04.01- Introduction to transport protocols
04.02- Distribution of multimedia
04.03- streaming
04.02- Streaming Protocols

Multimedia in rete

Maurizio Maffi

ISTI Information Science and Technology Institute

Applicazioni multimediali

Le applicazioni multimediali possono essere classificate nelle seguenti categorie:

1. Real time
2. Synchronous (interactive) / Asynchronous
3. Unicast / Multicast / Broadcast
4. One-way / bidirectional

Applicazioni multimediali

Real time
Sostanzialmente un sistema in tempo reale è un sistema che deve reagire ad un dato stimolo entro un tempo predefinito.

Hard time
i sistemi ‘hard’ richiedono un rispetto rigido dei vincoli di precisione temporale, in quanto mancare una scadenza significherebbe invalidare il funzionamento dell’intero sistema

Soft time
I ‘soft’ si limitano ad un rispetto statistico dei vincoli che, se forzati, portano ad una degradazione dell’applicazione che può però essere tollerata in funzione del suo costo per l’utilizzatore


Applicazioni multimediali

Unicast: il destinatario della tramissione è un unico host

Multicast: il destinatario è rappresentato da un gruppo di host

Broadcast: il destinatario della trasmissione è rappresentato da ogni host raggiungibile

Applicazioni multimediali - Multicast

Il **multicast** è un metodo per la distribuzione di dati verso un certo numero di host.

I dati vengono trasmessi solo una volta dall'host sorgente verso le destinazioni, che possono anche trovarsi su diverse reti.

Tipicamente, il gruppo di destinatari in una trasmissione multicast è dinamico, nel senso che è sempre aperto all'entrata di nuovi host o all'uscita di quelli già esistenti.

Questo dinamismo consente al sistema di operare anche in caso di interruzione nei normali servizi di rete.

Reti di computer

Per una migliore comprensione delle architetture, dei protocolli e tecniche, è necessario introdurre alcuni concetti di base relative alle reti di computer (in particolare a rete Internet).

Il modello ISO/OSI, concepito per reti di telecomunicazioni a commutazione di pacchetto, è costituito da una pila (o stack) di protocolli attraverso i quali viene ridotta la complessità implementativa di un sistema di comunicazione per il networking.

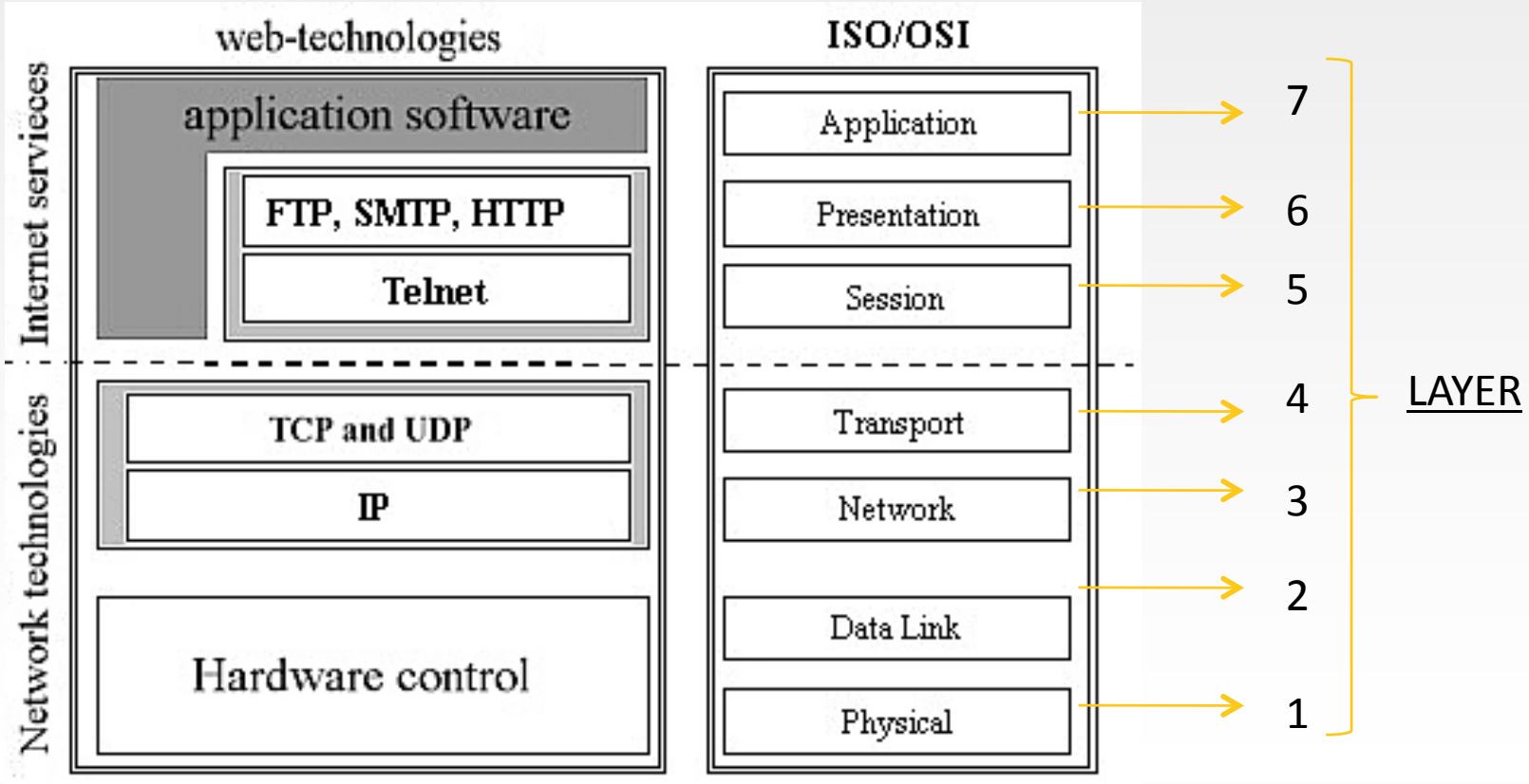
In particolare ISO/OSI è costituito da strati (o livelli), i cosiddetti layer, che racchiudono uno o più aspetti fra loro correlati della comunicazione fra due nodi di una rete. I layers sono in totale 7 e vanno dal livello fisico (quello del mezzo fisico, ossia del cavo o delle onde radio) fino al livello delle applicazioni, attraverso cui si realizza la comunicazione di alto livello.


Reti di computer

Ogni layer individua un protocollo di comunicazione del livello medesimo.

ISO/OSI realizza una comunicazione per livelli, cioè dati due nodi A e B, il livello n del nodo A può scambiare informazioni col livello n del nodo B ma non con gli altri: ciò conferisce modularità al sistema e semplicità di implementazione e reimplementazione.

Layers


web-technologies
ISO/OSI
Internet servieces
Network technologies
application software
FTP, SMTP, HTTP
Telnet
TCP and UDP
IP
Hardware control
Application
Presentation
Session
Transport
Network
Data Link
Physical
7
6
5
4
3
2
1
LAYER

Layer 1 physical

L’obiettivo di questo livello è quello di trasmettere bit su un canale di comunicazione.

Gli aspetti di progetto sono:

- Garantire che la comunicazione avvenga in modo corretto (es. se viene inviato un 1, venga ricevuto un 1 e non uno 0)
- Gestione esplicita delle caratteristiche meccaniche, elettriche e procedurali delle interfacce di rete (componenti che connettono l'elaboratore al mezzo fisico) e le caratteristiche del mezzo fisico

Si gestiscono, ad esempio:

- Tensioni scelte per rappresentare 0 ed 1
- Durata (in microsecondi) di un bit
- Trasmissione simultanea in due direzioni oppure no
- Forma dei connettori


Layer 2 data link

Obiettivo: far sì che un mezzo fisico trasmissivo appaia, al livello superiore, come una linea di trasmissione esente da errori di trasmissione non rilevati.

Trasmette *frame con "sufficiente"* affidabilità tra due entità direttamente connesse, rileva errori di trasmissione e (raramente) li corregge.

Operazioni legate ai frame
– delimitazione, ordinamento dei bit, suddivisione in campi, indirizzi, etc.

Rilevazione e correzione errori
– codici auto correggenti, ritrasmissione, basata su pacchetti ACK (*acknowledgement*)

Layer 2 data link

Funzionamento

- Spezzetta i dati provenienti dal livello superiore in frame;
- Invia i frame in sequenza
- Aspetta un acknowledgement (**ack**) per ogni frame inviato

Layer 2 data link

Compiti:

- Aggiunta di delimitatori (framing) all'inizio ed alla fine del frame;
- Gestione di errori di trasmissione causati da:
  - Errori in ricezione
  - Perdita di frame
  - Duplicazione di frame (da perdita di ack)
  - regolazione del traffico (per impedire che il ricevente sia "sommerso“)
  - meccanismi per l'invio degli ack
  - piggybacking (da pickaback, cioè trasportare sulle spalle) dei pacchetti ACK

Per le reti broadcast che devono controllare l'accesso condiviso al canale trasmissivo,è stato implementato uno speciale sottolivello del livello data link, il MAC (Medium Access Control)

Layer 3 network

Obiettivo: gestire l'instradamento di frame attraverso sistemi intermedi, e trovare percorsi alternativi in caso di problemi.

E’ il primo livello che si occupa del corretto instradamento (routing) dei pacchetti nella rete.

Il layer determina se e quale sistema intermedio deve essere attraversato dai messaggi al fine di giungere a destinazione.

A questo scopo il layer utilizza apposite routing tables per consentirgli di individuare strade alternative in caso si presentino degli errori (fault tolerance).

Layer 4 Transport

Obiettivo del layer 4 è quello di garantire una trasmissione affidabile, ottimizzando l'uso delle risorse e di fatto ignorando la topologia della rete o la presenza di ulteriori sistemi di routing intermediari (*end to end*).

Affidabilità: tutte le trame arrivano a destinazione, in copia unica e in ordine.

Ottimizzazione: traffico ripartito sui canali disponibili, prevenzione della congestione della rete.

Deve accettare dati dal livello superiore, spezzettarli in pacchetti, passarli al livello rete ed assicurarsi che arrivino alla *peer entity* corrispondente.

Layer 4 Transport

Si occupa della creazione di connessioni di livello rete:

1. Una connessione rete per ciascuna connessione trasporto
2. Molte connessioni rete per una singola connessione trasporto
3. Una singola connessione rete per molte connessioni trasporto , con meccanismi di multiplexing

Offerta di servizi al livello superiore:

1. canale punto a punto affidabile, che consegna dati in ordine e senza errori (il servizio più diffuso, *connection oriented)*
3. invio di messaggi isolati, con o senza garanzia di consegna (*connectionless)*
4. broadcasting di messaggi a molti destinatari (*connectionless)*

Layer 5 Session

Obiettivo: gestire il dialogo *end-to-end tra due* programmi applicativi che devono comunicare

Dialogo

- garantire la mutua esclusione nell'utilizzo di risorse condivise (*token*)
- intercalare domande e risposte garantendo la consequenzialità

Sincronizzazione

- stabilire punti intermedi (*checkpoint*) nella comunicazione rispetto ai quali entrambe le parti abbiano la garanzia che quanto accaduto prima sia andato a buon fine

Layer 6 Presentation

Obiettivo: gestire la sintassi dell'informazione lungo l'intero percorso *end-to-end, convertendo i vari formati* (crittografia, compressione, ri-formattazione).
Tre diverse semantiche consentono di definire le trasformazioni dei dati provenienti dal livello 7:

Sintassi astratta
– definizione formale dei dati scambiati dagli applicativi

Sintassi concreta locale
– come i dati sono rappresentati sui singoli sistemi

Sintassi concreta di trasferimento
– come i dati sono rappresentati lungo il percorso

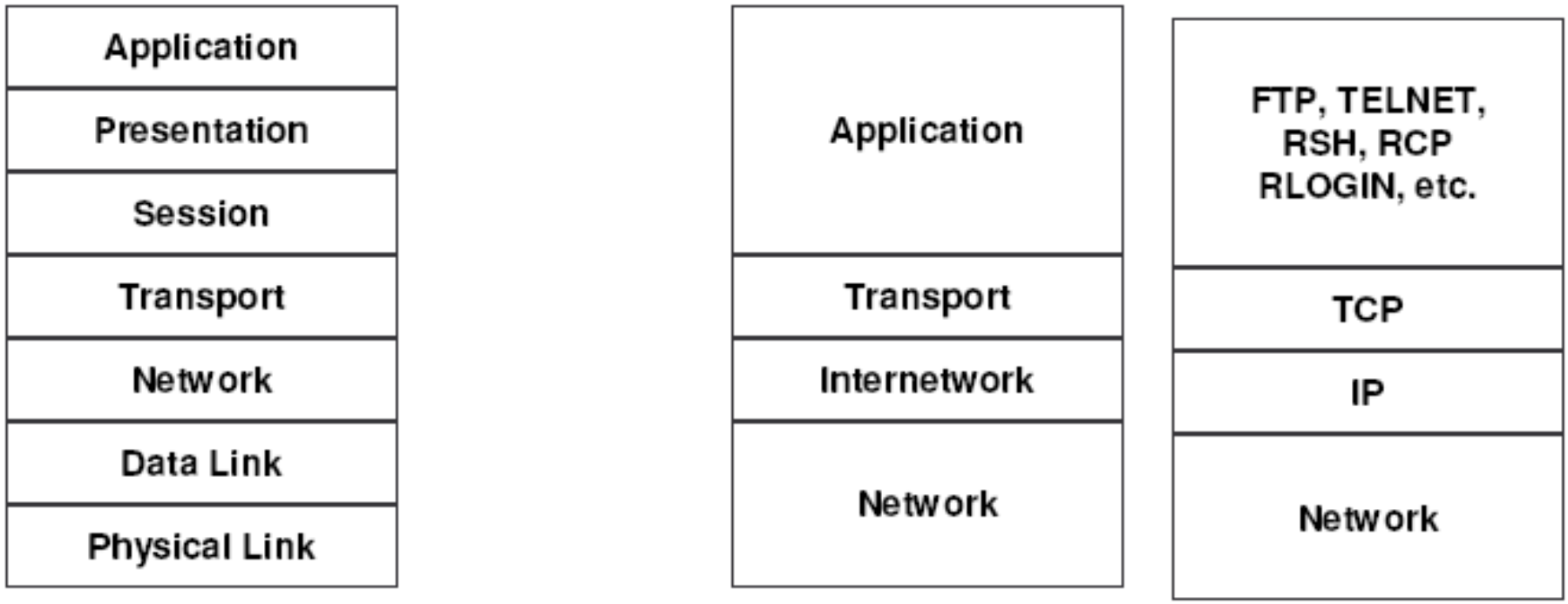
Layer 7 Application

Obiettivo: definire i servizi attraverso cui l'utente (non necessariamente umano) utilizza la rete, con tutte le relative interfacce di accesso

Servizi di utente
– terminale virtuale, trasferimento di file, posta elettronica, servizi di directory, etc.

Servizi di sistema operativo
– risoluzione di nomi, localizzazione di risorse, sincronizzazione degli orologi tra sistemi diversi, controllo di diritti di accesso, etc.

Protocolli internet e relazioni con iso/osi



Il protocollo IP

il protocollo IP (*Internet protocol*) che definisce il livello di rete (layer 3) di internet è:

1. NON ORIENTATO ALLA CONNESSIONE (connectionless): ogni pacchetto contiene l'indirizzo di partenza e di destinazione e può seguire un percorso diverso (routing).

2. NON AFFIDABILE: (best effort) ogni singolo pacchetto può non arrivare, o arrivare con un ritardo impredicibile a priori.

Il livello di trasporto

Nella suite protocollare del livello di trasporto (layer 4) sono disponibili due diversi protocolli di trasporto

1. **TCP (Trasmission Control Protocol):** è un protocollo con connessione di tipo affidabile, ossia tutti i pacchetti spediti arrivano rispettando l'ordine di arrivo.

2. **UDP (User Datagram Protocol**): è un protocollo senza connessione ed è di tipo non affidabile poiché i pacchetti spedito possono arrivare in ordine diverso o nel caso peggiore non arrivare affatto.

TCP vs UDP

Possiamo riassumere le differenze tra TCP ed UDP nel seguente modo:

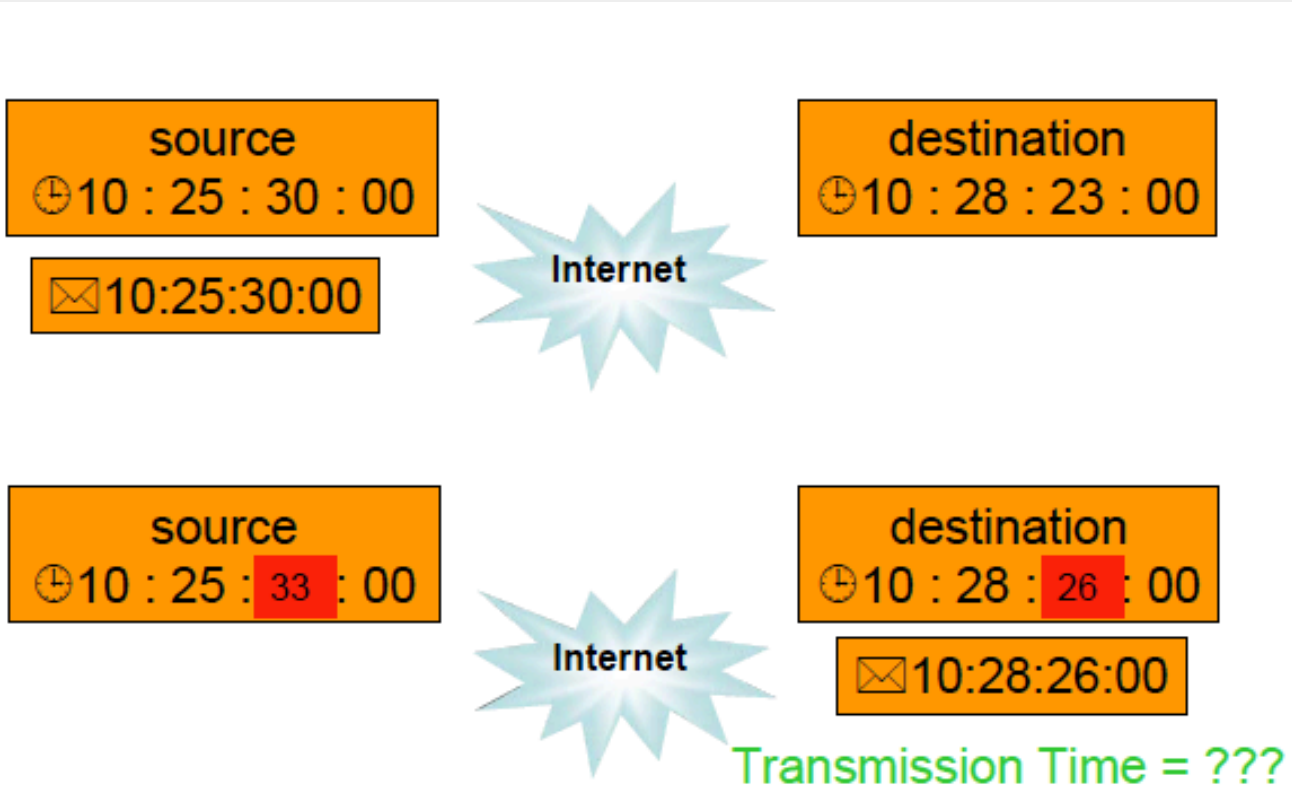
TCP	UDP
Garantisce la consegna ordinata dei pacchetti	NON garantisce la consegna ordinata dei pacchetti
Richiede più banda e più tempo di comunicazione a causa dei controlli per garantire la consegna	Non dovendo effettuare controlli per garantire le consegna dei pacchetti, risulta essere veloce.

Tempo di trasmissione

Calcolare il tempo necessario per trasferire un pacchetto da una sorgente verso la sua destinazione può, nella rete, essere un'operazione molto complessa.

Una delle cause è legata alla differenza di orario tra le due macchine, differenza che non può essere colmata con semplici operazioni di sincronizzazione.

Tempo di trasmissione

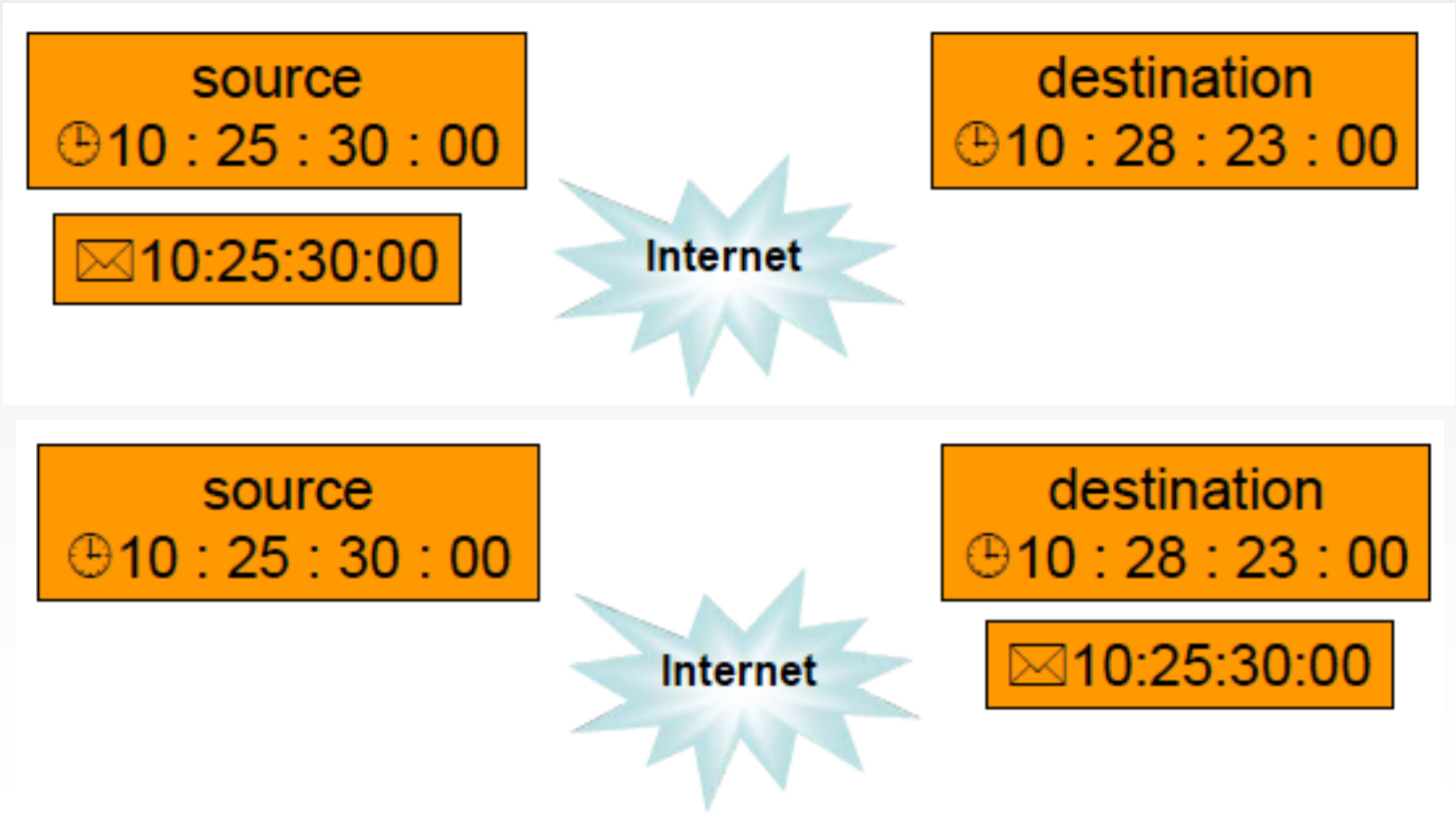


Round trip time

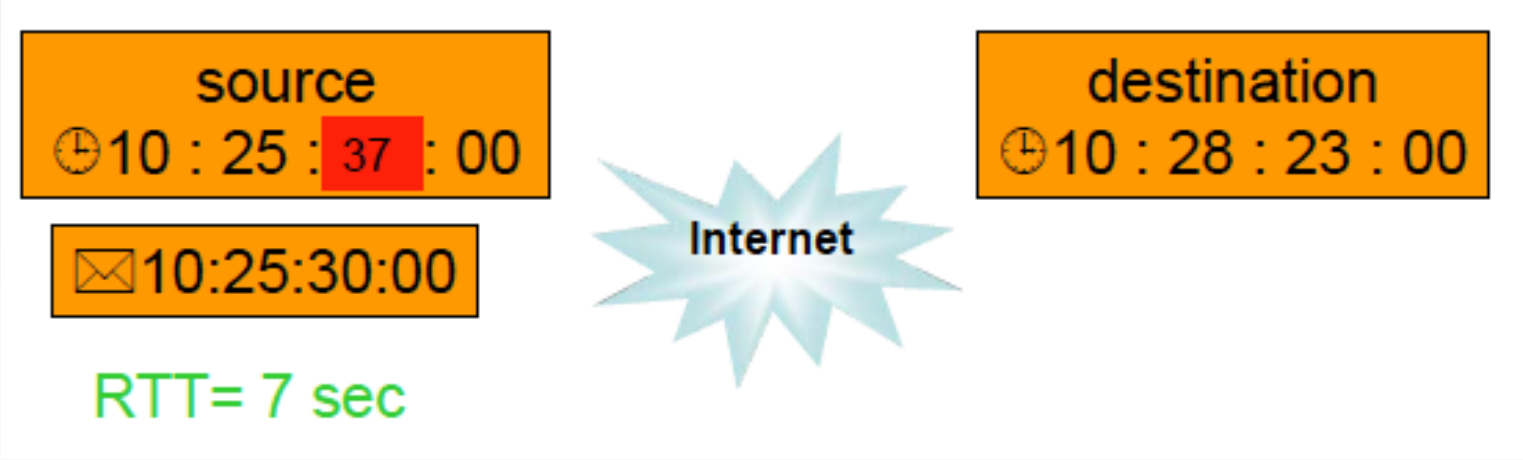
Al fine di misurare le performance di rete, è possibile utilizzare il Round Trip Time (RTT), come espressione del tempo necessario ad un pacchetto per giungere a destinazione e tornare indietro.

Questo tempo può quindi essere calcolato come differenza tra due diversi tempi ottenuti con il medesimo orologio.

Round trip time



Round trip time



Applicazioni multimediali

La scelta del protocollo di trasporto da utilizzare in ambito multimediale dipende fortemente dal tipo di applicazione multimediale che si vuole realizzare

- Possiamo distinguere due tipi di applicazioni

1. Applicazioni che richiedono il download completo delle risorse multimediali. Esempio: applicazioni peer-to-peer.

2. Applicazioni che NON richiedono il download completo delle risorse multimediali, ma operano su flussi generati dal media continuo nel tempo

Applicazioni multimediali

- Non sempre il download completo del file è una strategia utilizzabile ed esistono addirittura casi in cui ciò rappresenta una metodica di tipo insostenibile.

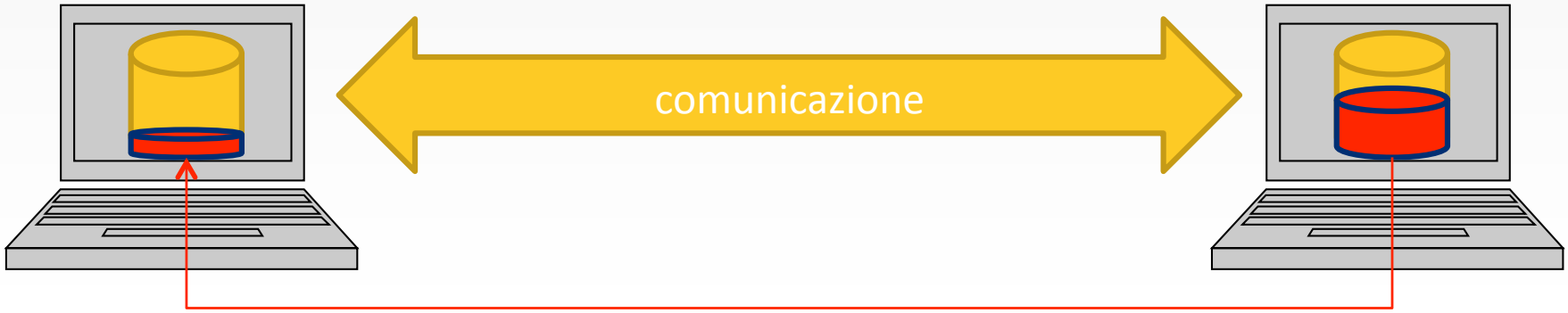
- Non è possibile, ad esempio, il download completo in caso di applicazioni che operino in tempo reale, come può essere per esempio la radio via Internet, o analogamente la TV via Internet.

- Esistono poi casi in cui il download sarebbe possibile ma sarebbe così lungo da scoraggiare l'utente. Per esempio nel caso del video on demand, il download di un intero film può richiedere anche ore e l'utente non è tipicamente disponibile ad attendere così tanto per iniziare a fruire del servizio che ha richiesto.

STREAMING

- quando il download sarebbe troppo lungo allora si utilizza una tecnica nota come *STREAMING* che opera considerando il media come stream , ovvero flusso , e non come file.

- *L'host ricevente* comincia il playout del media continuo (audio e video) prima che si concluda la comunicazione con *l'host trasmittente*



Streaming

- Il meccanismo di base dello streaming lo si può semplificare nel seguente modo:

➢ L'host sorgente inizia la trasmissione

➢ L'host destinazione riceve il primo pacchetto dalla sorgente e **aspetta** un certo periodo durante il quale accumula pacchetti che arrivano a destinazione

➢ terminato il periodo d'attesa l'host di destinazione comincia il playout dei pacchetti accumulati

➢ La trasmissione continua con la modalità descritte sino al termine

Streaming su UDP

- Tipicamente i sistemi di streaming operano su **UDP**, cioè il protocollo di trasporto che NON controlla l'integrità della trasmissione né l'ordine di arrivo dei pacchetti (*connectionless*).

- **PRO:** viene velocizzata la trasmissione liberandola dai controlli

- **CONTRO:** possono esserci pacchetti persi e pacchetti disordinati, causati dal servizio *best effort* (*massimo impegno*) offerto da IP.

Jitter

- Se il controllo di ordine ed integrità non è effettuato a livello di trasporto (UDP) allora deve essere effettuato a livello applicazione .

- Il disordine che si produce presso la stazione ricevente è causato dal fatto che ciascun pacchetto impiega un tempo diverso a raggiungere la destinazione.

- La variabilità del ritardo (e conseguentemente la sua impredicibilità) prodotto dalla trasmissione è detta ***DELAY JITTER*** (variazione del ritardo)

Buffer

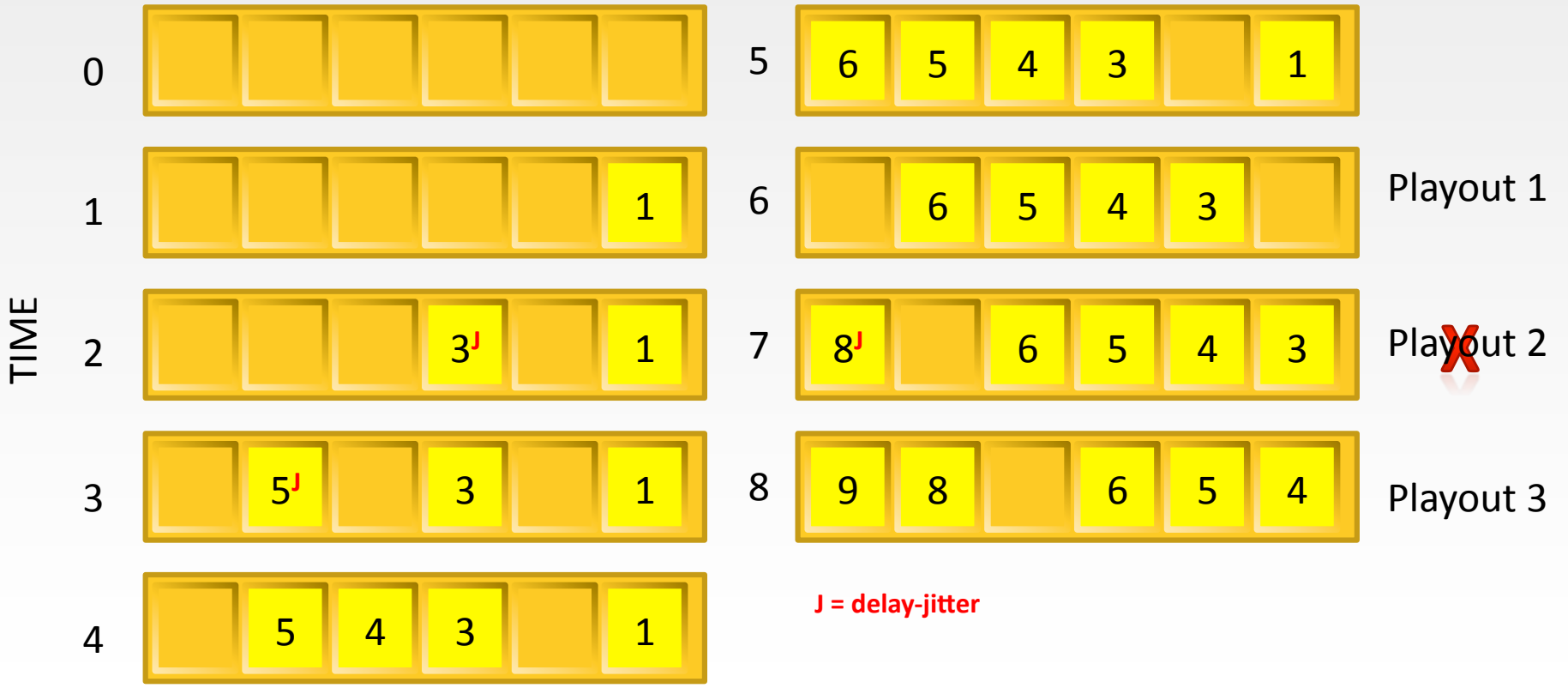
- L'host di destinazione deve predisporre uno spazio di memoria in cui accumulare i pacchetti in attesa del playout (*buffer*)

- Il buffer è una struttura a pila di tipo FIFO (first-in-first-out) che viene riempita da un lato e svuotata dall'altro

- Il buffer non si riempie in modo regolare

  - Perché i pacchetti possono arrivare disordinatamente
  - Perché alcuni pacchetti possono non arrivare affatto

Buffer


TIME
0
1
2
3
4
5
6
7
8
1
3J
1
5J
3
1
5
4
3
1
6
5
4
3
1
6
5
4
3
8J
6
5
4
3
9
8
6
5
4
Playout 1
Playout 2
Playout 3
J = delay-jitter

Packet Loss

- Nell'esempio precedente il pacchetto #2 non è arrivato in tempo per essere ascoltato
- Questo tipo di errore è detto **PACKET LOSS** e genera una interruzione del segnale
- Il pacchetto perso può essere
  - Realmente perso, ossia non arriverà mai al ricevente
  - Troppo in ritardo, arriverà al ricevente ma non in tempo per il playout

Buffer e QoS (Quality of Service)

La struttura a buffer ha sia dei vantaggi che degli svantaggi

- VANTAGGI:

  ➢ Effettua automaticamente il riordino dei pacchetti, tentando di compensare e gestire il delay jitter

- SVANTAGGI:

  ➢ I pacchetti messi in attesa ritardano forzatamente il momento del loro playout, continuando ad occupare spazio.
  Nell'esempio precedente il pacchetto 1 era presente al tempo 1, ma il suo playout avviene solo al tempo 6

Parametri di qualità

- I parametri per valutare la qualità di trasmissione audio/video su IP sono:
  - Il numero di pacchetti persi
  - Il ritardo tra il momento in cui la trasmissione parte ed il momento in cui il media viene visto o ascoltato
  - La variazione che questo ritardo può assumere a causa dell'impredicibilità del comportamento della rete (delay jitter)

Percezione

- La percezione che l'utente ha della qualità dello streaming è legata ai seguenti fattori
  - **Pacchetti persi:** può essere percettibile, come discontinuità il mancato arrivo del pacchetto
  - **Ritardo:** ritardo iniziale di buffering o attesa per rebuffering

Protocolli

Le applicazioni di streaming in ambito web sono tipicamente basate sui seguenti protocolli

Protocolli per le trasmissioni multimediali real time che usano anche TCP, ma preferibilmente **UDP**.

- RTP (Real Time Protocol)
- RTCP (Real Time Control Protocol)

Protocollo per il controllo dell'attività di streaming

- RTSP (Real Time Streaming Protocol)
  - Compressione e decompressione
  - Rimozione del jitter delay
  - Correzione degli errori

RTSP

Caratteristiche dell' RTSP

- Supporto per il controllo dell'erogazione in streaming del flusso da parte dell'applicazione (client – server)

- Supporto per il controllo del flusso da parte dell'utente (Rewind, stop, play ..)

- Protocollo di controllo "fuori banda". Vengono usate due connessioni, una per lo stream multimediale e una per il controllo.

Meta File

•La richiesta di attivazione di una connessione di tipo RTSP avviene tramite il web , con una richiesta di tipo HTTP che parte da un browser ed arriva ad un server, e quindi si avvia lo streaming tra client-server.

•Il browser richiede una risorsa che in realtà è un **meta file,** ossia un file che descrive il media di cui verrà fatto lo streaming.

•Il server web risponde inviandolo.

•Il browser lancia l'appropriato player e gli passa il meta file

•Il player seguendo le indicazioni del meta file avvia lo streaming