Capitolo 1
Introduzione

Trasmissioni Multimediali

Con il termine **multimedialità** s'intende il trasferimento di dati tramite un mezzo fornito di più canali (come una TV). *Multimedia is media and content that uses a combination of different content form. The term can be used as a noun (a medium with multiple content form) or as an adjective describing a medium as having multiple content form. The term is used in contrast to media which only use traditional forms of printed or hand-produced material. Multimedia includes a combination of text, audio, video, still images, animation, and interactivity content forms.*
Un testo è considerato multimediale se coordinato ad un video od un audio, i **dati multimediali** sono informazioni che vengono fruite in maniera multimediale; constano di:

1. differenti tipologie di contenuto;
2. concorrono a costruire una sola informazione (multimediale);
3. si combinano tra loro per colpire contemporaneamente più sensi.

Un esempio di contenuto multimediale è dato da una pagina web con link a risorse esterne, magari con un audio di sottofondo, qualche foto ed un video embedded. Tecnicamente una pagina web qualsiasi soddisfa completamente la suddetta definizione riguardo la natura del contenuto ed il suo formato, poiché presenta diverse tipologie di risorse combinate tra loro e perchè non è una forma tradizionale di comunicazione.
Nella lingua inglese il termine **media** considera un mezzo fisico, come un libro od una rivista, che contiene informazioni → *multimedia* è allora un insieme di mezzi fisici distributori di informazioni; il termine stesso indica che si ha a disposizione interattività momento per momento con una possibilità di interazione diretta ed attiva con la trasmissione multimediale in quanto tale.
La definizione di **realtime** e di multimediale non coincidono. La differenza tra Youtube ed una pagina web qualsiasi è una questione prettamente filosofica (perchè dal punto di vista tecnico non ci sono differenze) → Youtube riesce a trasformare la fruizione delle informazioni in maniera non lineare. Ci si aspetta però da un contenuto multimediale che sia anche coerente, interattivo e realtime. Il problema da risolvere è: quando i dati devono essere fatti arrivare all'utente? Nella **fruizione non lineare** i dati non sono più fruiti dal primo all'ultimo (o dall'inizio alla fine) ma è possibile saltare ad un nuovo punto d'interesse senza prima arrivare “in fondo alla pagina” (è come pescare da un contenitore di informazioni quella che ci interessa in quello specifico momento senza l'obbligo di dover leggere tutte le informazioni del contenitore per poter saltare da un contenuto all'altro senza problemi) → **hypermedia**, ovvero un ipertesto senza il testo (per esempio un video ricco di collegamenti che da ogni elemento di ogni frame puntano ad un altro contenuto). Wikipedia e Youtube sono gli esempi più vicini e simili al concetto di hypermedia anche se non completamente calzanti. Si vorrebbe avere, inoltre, un sistema automatico di classificazione delle informazioni che esuli dai feedback degli utenti (come avviene per Youtube per la “fama” di un video)
Così però non va bene perchè quello che vorremmo avere è un'interattività che

assomigli ad un hyperlink su una pagina web. Il media vorrei che mi desse la possibilità di indicare un oggetto in un filmato. Quando per esempio si visiona un video di una vecchia Fiat 500 si vorrebbe poter cliccare sulla sua immagine per poter accedere alle informazioni ad essa relative: tuttavia questo è un procedimento assai difficoltoso dovuto innanzitutto alla cultura d'appartenenza del fruitore del contenuto. Gli ostacoli principali comunque sono dati dal dover definire degli standard accessibili per chi eroga i contenuti (che ad oggi non esistono) e "spiegarlo" a coloro che creano contenuti; il tutto per guadagnarci su investendoci tuttavia il meno possibile. La fruizione non lineare oggi non fa parte dei contenuti multimediali strettamente tali, perlomeno non in maniera semantica, tant'è che si usano dei canali separati per la fruizione ed il coordinamento delle informazioni (uso il browser per comunicare al sito Youtube che voglio vedere un video diverso). Al massimo possiamo aggregare tutti i contenuti provenienti da canali diversi potendo così saltare dall'uno all'altro.
La **fruizione coordinata** si ha quando abbiamo dei flussi di dati da sottoporre all'utente provenienti da canali chiaramente differenti e questi devono essere obbligatoriamente coordinati tra loro per poterne fruire in maniera corretta e godibile (per esempio i sottotitoli di un programma RAI dati dal televideo che non è parte dell'immagine, tuttavia le "parole" appaiono nel momento giusto). Usando socket diverse bisogna tener conto dei ritardi di trasmissione che possono essere presenti in maniera discontinua sulle due socket perdendo in sincronia. Il televideo sfrutta l'ampiezza della trasmissione per trasmettere assieme alle immagini informazioni aggiuntive che normalmente non vengono visualizzate. Quando i televisori erano a tubo catodico l'immagine veniva "dipinta" sullo schermo da un fascio di elettroni e lo spazio tra una proiezione e l'altra era riempito dal televideo. Il coordinamento dei flussi oggi è completamente demandato ai formati di memorizzazione. Un container come l'AVI (file contenente metadati che coordina i flussi) struttura al suo interno una serie di flussi che possono essere video, audio (più di uno) che vengono coordinati tra loro tramite dei marcatori temporali: all'istante T tutti i flussi si trovano ad un certo punto dello stream di byte (del file contenitore). Con contenuti più complessi è molto più difficile anche perchè non è sempre detto che questi siano facilmente sincronizzabili. Si sfrutta molto la sincronia temporale (per tempo) ma risulta più difficile sfruttare quella di tipo contenitivo (per contenuto).
La **fruizione interattiva** indica che il contenuto debba evolvere in base all'interazione con l'utente.

Esempio. *MYST, gioco creato nel 1993 da Apple con un grosso file Quicktime voleva proporre una tecnologia interattiva. Si aveva una serie di fotogrammi che si susseguivano; nel rimaneggiamento del suddetto gioco ai fotogrammi si è sostituito un filmato per regione.*

Per ottenere un contenuto davvero interattivo serve comunque un contributo personale differente da apportare all'informazione e non si può proporre lo stesso contenuto a più utenti utilizzando un unico contenitore d'informazione. Tuttavia la mole di dati che si gestiscono potrebbe essere grossa e risulta difficile utilizzare un supporto di memoria limitato → i dati non sono più in locale ma sfruttano la rete e arrivano ondemand all'occorrenza. Se al server si connettono tantissimi utenti contemporaneamente (come d'altronde è auspicabile) la rete

potrebbe non reggere più il carico. Si trovano allora dei compromessi, come erogazione di video ed audio di qualità più bassa per alleggerire il flusso di dati senza perdere però qualità abbastanza da riuscire a fruire comunque del contenuto; si può anche stabilire l'ordine in cui far arrivare i dati. A livello software è possibile sfruttare tecniche che trovano il giusto compromesso tra quello che si può avere in locale e ciò che arriva dalla rete (in alternativa, in quale ordine far arrivare i dati per rendere il contenuto fruibile). Qualche volta la fruizione dovrebbe essere realtime, si pensi ad una telefonata od ad una videoconferenza, ma solo nel caso in cui si abbia a che vedere con un essere umano (in caso contrario non ha senso parlare di realtime).
Un **sistema realtime** è un ambiente in cui il tempo di risposta di chi eroga una informazione ha lo stesso ordine di grandezza del tempo impiegato dall'utente nel fruire dell'informazione. Si divide in:

- *soft realtime (mission critical)*: l'informazione sarà disponibile entro il tempo T nell'x% dei casi;
- *hard realtime (life critical)*: l'informazione sarà sicuramente disponibile entro il tempo T.

L'assenza o la presenza di realtime non si nota quando c'è disincronia di due canali che, in diretta, non sono coordinati: per il nostro cervello infatti è più importante cogliere l'audio rispetto al video. Si dicono **trasmissioni realtime** quei trasferimenti d'informazione nei quali un utente è in grado di interagire con un sistema realtime senza che questo perda le sue caratteristiche di sistema realtime. Se poi ci sono due utenti anziché un sistema client-server, la cosa non cambia molto. Per sfruttare il realtime bisogna creare delle infrastrutture apposite: si devono commutare i circuiti e creare reti dedicate, ed è molto costoso. Per ottenere un hard realtime si sfruttano degli accorgimenti hardware di questo tipo. Per il soft realtime invece sfrutta accorgimenti software per agevolare il realtime, nei sistemi operativi e nelle infrastrutture di rete, come fare scheduling dei pacchetti con alte priorità anche se non è banale. Tutto ciò aveva senso finchè le risorse avevano un prezzo considerevole ma oggi non è più un problema: la banda non costa più nulla (o quasi) e si preferisce fare *overprovisioning*.
Oggi il realtime non è fisicamente possibile finchè si userà una rete best effort come quella di internet, tuttavia mandiamo i dati e spesso andrà bene così. Dopotutto, Youtube funziona, no?

Capitolo 2
Codifica e Compressione

Codifica e Compressione Multimediale

Codificare vuol dire stabilire un formato digitale (senza non si va da nessuna parte); una forma di codifica è data dallo stabilire una mappa tra informazione e bit da usare per tradurre tale informazione. Ma come passare da informazioni analogiche ad informazioni digitali? Una volta che un'informazione viene digitalizzata può asservire ad una quantità di scopi: si possono nascondere informazioni all'interno di un immagine senza che l'occhio umano se ne renda conto; si possono applicare sistemi crittografici; ecc. Sistemi di codifica fatti come si conviene possono offrire una buona protezione da eventuali errori di trasmissione dell'informazione.
Spesso si associa il termine **codifica** a quello di **compressione**, ma non sono la stessa cosa (anche se una *codifica* che risparmia spazio si tende a chiamarla *compressione*): una codifica tende a ridurre al numero minimo di bit l'informazione a patto che questa sia ancora fruibile ed intellegibile. La compressione di un contenuto è un effetto collaterale di una codifica ben strutturata.

Esempio. *Quando lavoro con un video codifico il filmato in una sequenza di frame → ogni frame viene codificato a sua volta in una matrice di pixel, ciascuno dei quali è tradotto in formato RGB (3 byte) → il tutto producendo un'informazione di grosse dimensioni (lo spazio occupato corrisponde a 3*640*480*25*60 = 1.28 GB). Con la compressione riduco la necessità di occupare risorse (specialmente su disco) in termini di spazio e di tempo di trasferimento; inoltre elimino tutte le informazioni non necessarie tra quelle non percepite dall'utente umano e quindi non fondamentali per la fruizione del contenuto.*

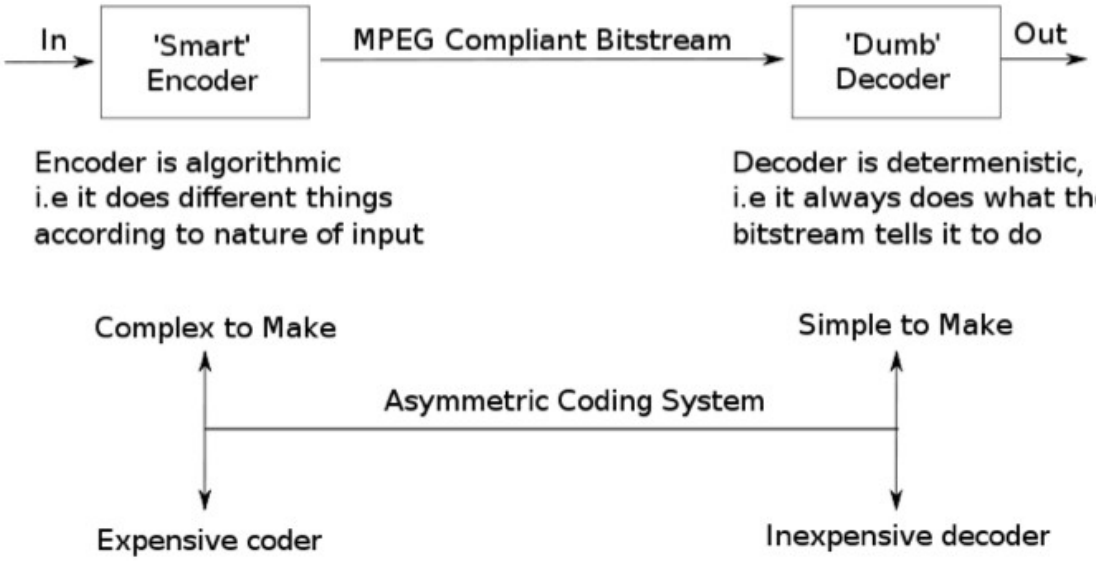
Esistono tecniche indipendenti dal contenuto per creare delle associazioni tra un insieme da rappresentare e stringhe binarie, fatte di bit, per la codifica tali per cui il numero totale di bit utilizzati è il minimo possibile (codifica di Huffman → metodo che codifica in maniera più stringata le informazioni più frequenti o con le occorrenze più alte all'interno del file) dette anche **codifiche lossless**: queste tecniche non fanno perdere informazioni e riducono solo lo spazio totale occupato. Una **compressione lossy** invece avviene quando un sistema di codifica tenta di scartare le informazioni inutili o che possono essere ricostruite in maniera automatica, agevolando così la compressione al massimo delle possibilità. Si ha **ridondanza spaziale** quando all'interno di un video o di un contenuto si hanno scene simili molto vicine (quindi informazioni simili tra loro aggregate) tanto da creare "macchie" dello stesso colore in una immagine. La **ridondanza temporale** si ha quando i contenuti che evolvono nel tempo non variano enormemente in due istanti successivi: in un video con una persona che parla non ho bisogno di mandare sempre anche lo sfondo. La ridondanza spaziale cambia aspetto a seconda se si trova sul piano orizzontale o verticale.
Siamo abituati a vedere il risultato di una operazione di **compressione** come

un “prodotto scadente” , tuttavia è importante capire a che uso viene destinato il prodotto di una compressione: un'immagine compressa per essere usata miniaturizzata non stona rispetto alla stessa, a dimensioni reali, non compressa. In una immagine sono importanti i bordi ed i contorni delle figure, meno lo sono i colori: l'occhio ed il cervello sono infatti sensibili alle alte frequenze (scaturite da cambi repentini di forme e colori) e meno alle basse.
La compressione è una questione complessa perchè dipende da parametri non direttamente misurabili e che sono difficili da controllare come lo è il mezzo della fruizione del contenuto, dipendentemente da agenti fisici o psicologici. Il mezzo di fruizione impone limiti tecnologici non evitabili; si rischia così di codificare delle informazioni che potrebbero non essere mai fruite. Il nostro cervello per primo non da la stessa importanza a tutte le informazioni e ricostruisce autonomamente quelle parti di informazione che non sono giunte o che ha scelto di non cogliere, è inoltre sensibile a variazioni istantanee visive ed uditive (individua più facilmente le forme e siamo più sensibili alle variazioni istantanee di un suono). La ricostruzione dell'informazione è fattibile comunque entro una certa soglia di tolleranza (che vale per ciascuno dei 5 sensi) e nella codifica è meglio risparmiare sul video anziché sull'audio. I parametri psicologici derivano ovviamente dalla nostra cultura ed il cervello si ricorda le informazioni a lui più congeniali (il nostro cervello riconosce automaticamente forme ed oggetti in base alla nostra esperienza, in maniera totalmente istintiva facendo sì che ci ricordiamo solo degli “aspetti importanti”): anche nei contenuti digitali è possibile ricostruire le informazioni mancanti, ed il cervello ricostruisce la realtà come la vorremmo e non come realmente è.

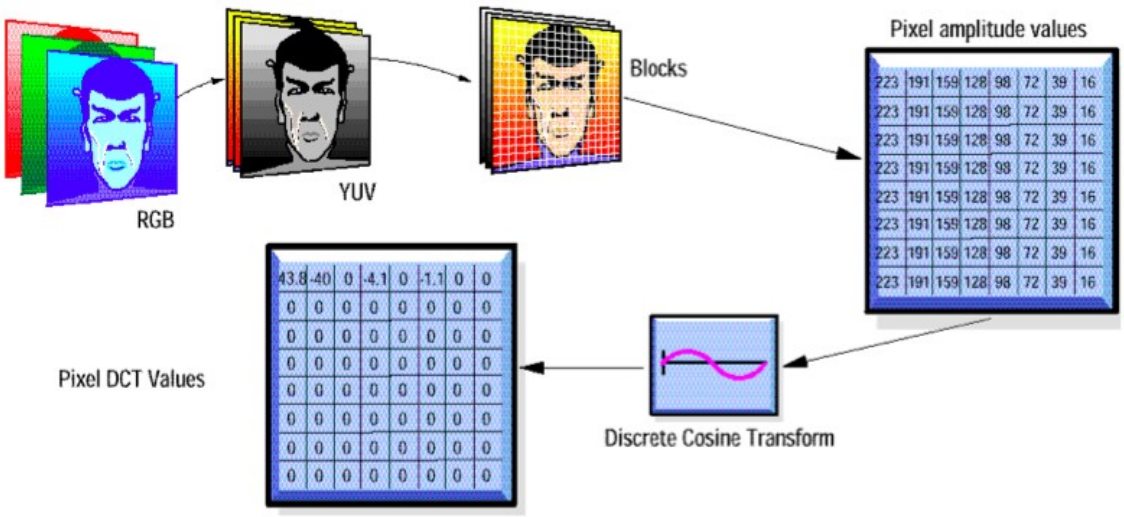
Audio e Video

Audio e **video** sono concetti chiaramente differenti: si tratta di tipi di dati multimediali profondamente diversi e vanno trattati in maniera disgiunta. Hanno differenti vincoli tecnici dovuti a frequenza e campionamento, fisiologici dati da sensibilità non uniforme o diversamente distribuita e psicologici dati dalla diversa tolleranza rispetto la perdita di dati (nell'audio, quando si perdono dati si tenta un'interpolazione che spesso degrada la qualità dell'audio stesso).
MPEG (*Moving Picture Experts Group*)→ È una famiglia di standard che permette di fare codifica ma non è “la soluzione a tutti i mali”: definisce criteri di codifica e compressione ed i suoi sotto-standard definiscono sotto-regole per la compressione e la codifica in specifiche situazioni (es: sul web). Si sono date delle regole per la compressione spaziale e temporale in maniera tale da non disturbare troppo la percezione dell'utente medio.
MPEG è un'organizzazione fondata dalla ISO nel 1988 ed ha il compito di definire degli standard per la compressione di audio e video, tant'è che di versioni di MPEG ne esistono molteplici. È curioso sapere che il problema sollevato dal MPEG è sorto ancor prima che si presentassero l'esigenza e la tecnologia adeguate per porlo, il problema. MPEG-1 era riservato al compact disc (ormai in disuso); MPEG-2 a DVD e Digital Video Broadcasting satellitare (con le diciture *Program Stream* e *Transport Stream* si distinguono le modalità di lettura dei pacchetti); MGEP-4 era dedicato al DVB satellitare (ancora un po' legato a MPEG-2), via cavo e terrestre ed al Blue Ray Disc.
Le trasmissioni televisive maggiormente compresse sono i cartoni animati.
La codifica MPEG è volutamente asimmetrica perchè in questa maniera è giustificata la fatica a livello di tempo e di risorse nel codificare il contenuto la

prima ed unica volta a patto di una decodifica rapida e leggera senza bisogno di impiegare un hardware dedicato; inoltre, se si cambia tecnologia di codifica anche una sola volta non si è costretti a cambiare anche tutte le decodifiche a lato client. Lo standard detta solo le linee guida per la creazione di un byte-stream ed è il codificatore a fare tutto il lavoro pesante per poter creare uno stream corretto così da lasciare al decodificatore il lavoro meno oneroso. Tenendo fisso lo standard è possibile far evolvere parallelamente i due estremi: si può implementare codificatori più efficienti senza modificare il software di visualizzazione e si possono creare nuovi player (e nuovi dispositivi) senza codificare nuovamente i contenuti.
MPEG Compliant Bitestream → si hanno delle regole con le quali i bit viaggiano da un server ad un client e sono regole assolute che non possono essere modificate perchè sono regole sulla costruzione del flusso di dati.

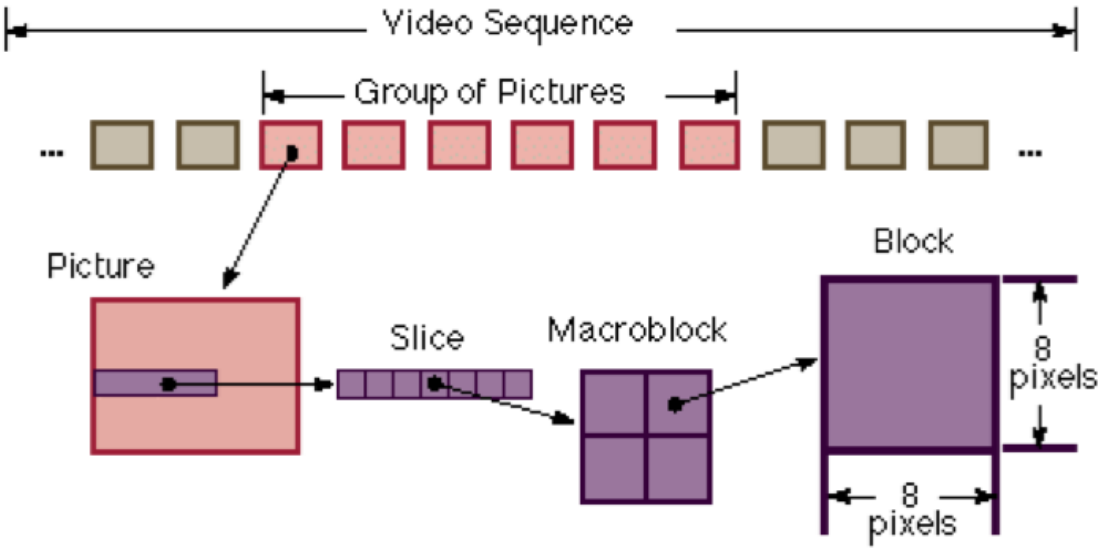


MPEG costruisce video codificando una sequenza di immagini (fotogrammi) e tenta di eliminare tutte le ridondanze spaziali: attenzione, non si tratta di una sequenza di immagini JPEG, perchè allora si parlerebbe di codifica MJPEG.



Una immagine RGB diventa quindi una immagine YUV e da una immagine

contenente informazioni sul colore si passa ad uno spazio in cui l'immagine contiene informazioni di *cromanza* e *luminanza* (alle quali il cervello, tra le altre cose, è più sensibile) allora l'informazione del colore RGB diventa sostanzialmente una sfumatura ed una illuminazione. Sul valore di luminanza è possibile sacrificare informazioni durante la codifica. Data la matrice YUV di un blocco tutti i suoi elementi vengono espressi per differenza rispetto al primo; successivamente si fa una riduzione ricampionando il segnale. A questo punto l'immagine viene suddivisa in una matrice di pixel alla quale si applica una *trasformata a coseno discreto DCT* che smacchia ed uniforma i dati quando i pixel tra loro vicini sono molto simili (l'obiettivo della DCT è di eliminare le basse frequenze che non vengono percepite come importanti dall'occhio umano): la matrice viene rappresentata in maniera intelligente: se i pixel si assomigliano i valori relativi a tali pixel presenteranno una media tra i valori dei singoli pixel mentre dove l'informazione manca si inseriranno un sacco di zeri (informazioni che si possono tranquillamente tralasciare ed eliminare: la DCT elimina così la ridondanza percettiva). Tale matrice dopo essere stata sottoposta alla DCT può quindi essere **linearizzata**: si lasceranno in testa alla linearizzazione tutti i dati ed i valori diversi da zero inserendo tutti gli zeri in coda (coda che poi verrà tagliata: il livello di "taglio" della DCT determina la qualità della codifica e la dimensione finale dei dati).

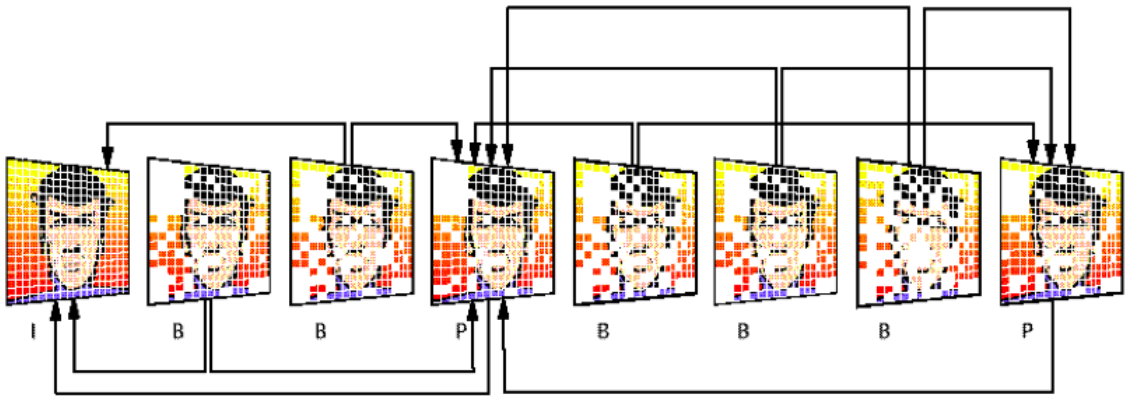


Data un'**immagine** la si taglia in **slice** orizzontali ognuna delle quali composte da **macroblocchi** che andranno codificati uno dopo l'altro. Un **GOP** ha, in testa, una immagine "fatta bene" con a seguire una serie limitata di immagini costituite da ridondanza spaziale dovuta alla differenza con la prima. Quando si comprime e si codifica si possono settare diverse informazioni tra cui il *bitrate* e la qualità dell'immagine di sintesi. La DCT possiede un fattore di taglio e per abbassare la qualità dell'immagine risultante si deve comunicare di gettare via più informazioni possibile. Di solito si fissa un numero minimo di GOP per frame e si decide il taglio della DCT.
Il **frame** è sostanzialmente un'inquadratura corrispondente ad una matrice di pixel che rappresenta l'immagine. Durante una trasmissione televisiva si hanno all'interno di ciascun frame due **semiquadri** per un totale di 25 frame che corrispondono a 50 semiquadri (al secondo) il tutto escogitato per rendere il movimento più fluido e per sfruttare l'alimentazione del dispositivo a 50 Hertz

come clock interno (negli Stati Uniti si usano 20 frame a causa della differente frequenza data dalla corrente dell'alimentazione). Si vuole tradurre il frame in una sequenza di bit il più contenuta possibile. Ma non basta perchè il problema della ridondanza temporale rimane (quello visto sin qui è il formato MJPEG): bisognerebbe ottimizzare la ridondanza. Per risolvere il problema si possono definire diversi tipi di frame:

1. **indipendenti** (detti *frame I*);
2. **predittivi** (*frame P*) che prevedono la variazione del tempo → tale variazione può essere trasmessa come differenza di immagini o come lo spostamento di un macroblocco all'interno dell'immagine.

Si prende in esame un frame e studiandolo si vogliono determinare alcune zone al suo interno che "vanno alla deriva" nei frame successivi (per capirci: le uniche differenze tra due immagini stanno nello spostamento di uno o più oggetti che semplicemente slittano di posizione senza cambiare nulla nel loro aspetto). È così che si costruisce una matrice **Motion Vector** → la compressione così funziona meglio perchè ogni tassello della matrice motion vector mostra lo spostamento di quell'area dell'immagine rispetto alla precedente. Tuttavia si delega al sistema di codifica di scegliere la tolleranza con la quale accettare o meno uno spostamento nell'inquadratura al fine di trovare le giuste somiglianze. Siccome tale ricerca impiega tempo si tenta di prendere l'area in esame abbastanza piccola.



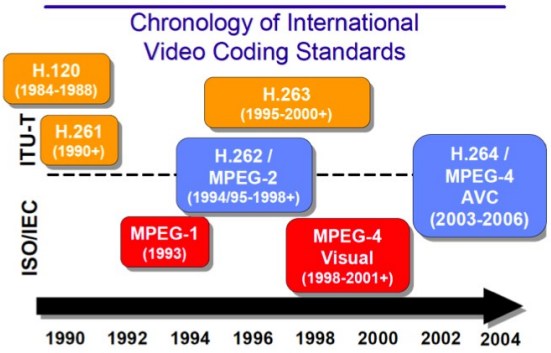
In ogni caso la **predizione** è la soluzione migliore. Esistono frame indipendenti codificati in JPEG e frame predittivi codificati in motion vector: è possibile costruire una motion vector anche guardando al futuro (e non solo al passato): si pensi ad un'inquadratura all'interno della quale passa un'auto e temporaneamente compare dietro un'albero, sarebbe sciocco dimenticare che l'auto ad un certo punto comparirà dall'altra parte. Si definisce allora un terzo tipo di frame detto **bidirezionale** (*frame B*) che descrive una scena per differenza con quelle sia precedenti che successive (risparmiando così ancora più bit). Quando si spediscono i GOP si manda un frame I poi un frame P per differenza, che è predittivo, così riesco a visualizzare anche altri due frame B che dipendono appunto dal frame I e dal P appena mandati; così facendo si perde un po' di tempo aggiungendo ulteriore ritardo anche se contenuto. Di frame I ne esiste uno per GOP (sostanzialmente il primo) al quale seguono opportuni frame P, quindi B a riempimento dipendenti da quel P:

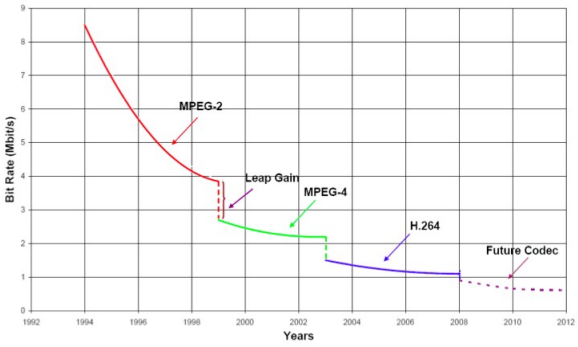
1. i *frame I* sono codificati indipendentemente come immagini JPEG-2000;
2. i *frame P* invece sono codificati per differenza rispetto al frame I che li precede (possono includere cambiamenti e vettori di movimento);
3. i *frame B* sono codificati come differenza con i frame I/P che li

precedono o li seguono (da qui il nome di predizione bidirezionale).

MPEG costituisce una famiglia di standard che ha conteso il territorio ad altri standard di codifica simili:

1. agli standard ITU-T prettamente ingegneristici che per primi hanno dato le linee guida complete per una codifica fortemente monolitica;
2. agli standard ISO/IEC, orientativamente più informatici, che davano solamente delle indicazioni generiche lasciando l'implementazione completamente libera;





L'evoluzione nel tempo da parte di queste codifiche ha portato ad un basso rapporto di compressione che è diminuito grazie alla continua scoperta di algoritmi più efficienti senza perdita di qualità e di informazioni importanti. Come spesso capita tra una tecnologia di codifica e l'altra è visibile un netto salto tecnologico-generazionale di massimo abbattimento di bitrate.

Continuando esattamente su questa strada si arriverà presto all'**entropia dell'informazione**, la quale misura quanto è prevedibile l'informazione che arriverà in futuro. Se l'entropia è alta non si saprà quale informazione sta arrivando, il chè comporta che si avranno pochissime informazioni a riguardo.

Capitolo 3
Quality of Service

Qualità di servizio

I dati multimediali codificati devono possibilmente arrivare a destinazione esattamente come serve per poterne usufruire (bella pretesa!).
Si definisce **qualità di servizio** quell'insieme di parametri istituiti per la garanzia deterministica sul comportamento del sistema; solitamente sono parametri come, per esempio, quello sulla *disponibilità del servizio* (espressa in percentuale su un tempo molto lungo). I dati devono arrivare a destinazione con delle garanzie.

Esempio. *Nel caso di un filmato video vorremmo che un frame arrivi ogni 40 millisecondi (meglio, ogni venticinquesimo di secondo) sempre e con esattezza, sennò il filmato andrà a scatti. Tuttavia non potremmo mai farci niente caso mai succeda (l'intero corso si basa sulla nostra incapacità di porre rimedio a questi disastri).*

Ci si appoggia ad Internet che per sua natura è una rete *best effort* (ovvero che fa del suo meglio per garantire che i dati viaggino ed arrivino a destinazione utilizzando i mezzi a sua disposizione, per inciso Internet deve il suo funzionamento più che accettabile proprio a questa sua caratteristica) e quindi non ci si può fare niente se per caso qualche frammento di informazione si perde "per strada". La qualità di servizio sarà sempre legata a quanto si paga per poter usufruire della rete.
Potendo esprimere qualche desiderio, si vorrebbe che i dati che arrivano comunque a destinazione siano fruibili e che quindi venissero garantite le risorse necessarie, un tempo massimo di consegna, una percentuale di pacchetti persi (nel peggiore dei casi), ed una buona qualità del servizio (che però non è tecnologicamente esprimibile) il tutto fruibile ed inteso come una buona qualità innanzitutto percettiva. Dal punto di vista tecnologico si vuole specificare dei parametri di comportamento, tutti completamente misurabili:

1. disponibilità della connettività;
2. banda trasmissiva, come numero di bit per secondo;
3. ritardo di trasmissione;
4. probabilità massima di perdita di dati;
5. livello di jitter accettabile (dove il jitter è una variazione istantanea sui tempi medi di arrivo delle informazioni). Il jitter è calcolabile ma non è importante a livello percettivo, tant'è che un video può tranquillamente essere velocizzato o rallentato fino al 10% senza che un essere umano se ne renda conto, l'importante è che non ci siano grosse variazioni temporanee.

Service Level Agreement

Uno **SLA (Service Level Agreement)** è la triste realtà con cui gli utenti delle telecomunicazioni devono aver a che fare. È un accordo informale nel quale si stabiliscono una quantità di termini di servizio che va tradotto poi in parametri tecnologici. Nulla dice che sia semplice: il compito del tecnico è quello di

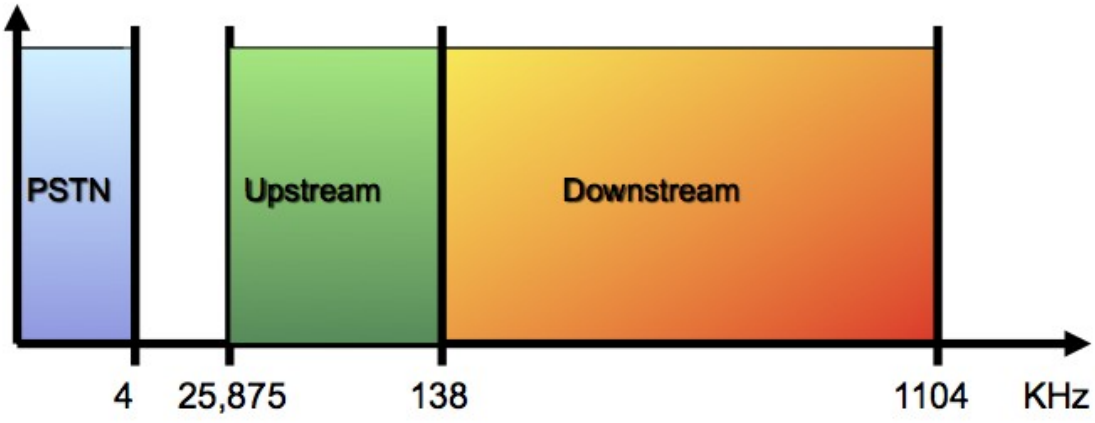
tradurre lo SLA in effettivi parametri tecnologici e nessuno MAI scriverà su un contratto che la probabilità di perdere dati sarà inferiore al 3%, al più diranno che sarà sufficiente telefonare.

Disponibilità di connettività. Si tratta della garanzia che la rete sia sempre e comunque disponibile dal punto di vista degli apparati e della reale disponibilità. La rete potrebbe essere funzionante ma può esserne inibito l'accesso (infatti quando si attiva un abbonamento dati GSM ci si pone sempre il problema se a "casa nostra prende"). I parametri certi son fissati per legge quando si parla di apparati telefonici per i quali un operatore deve garantire il 99,999% di disponibilità di rete e se per caso la qualità del servizio cala perde la licenza, ad un operatore mobile basta il 99% ma nulla è assicurato per la connettività dati (il telefono è un bene di prima necessità ma internet no, nessuna garanzia viene dall'internet service provider).

Banda trasmissiva. Si tratta della capacità da parte di un collegamento di smaltire i dati (per l'ADSL è una informazione che vale solo fino al D-SLAM). La fornitura di connettività è data dal provider che si appoggia ad una infrastruttura già esistente:

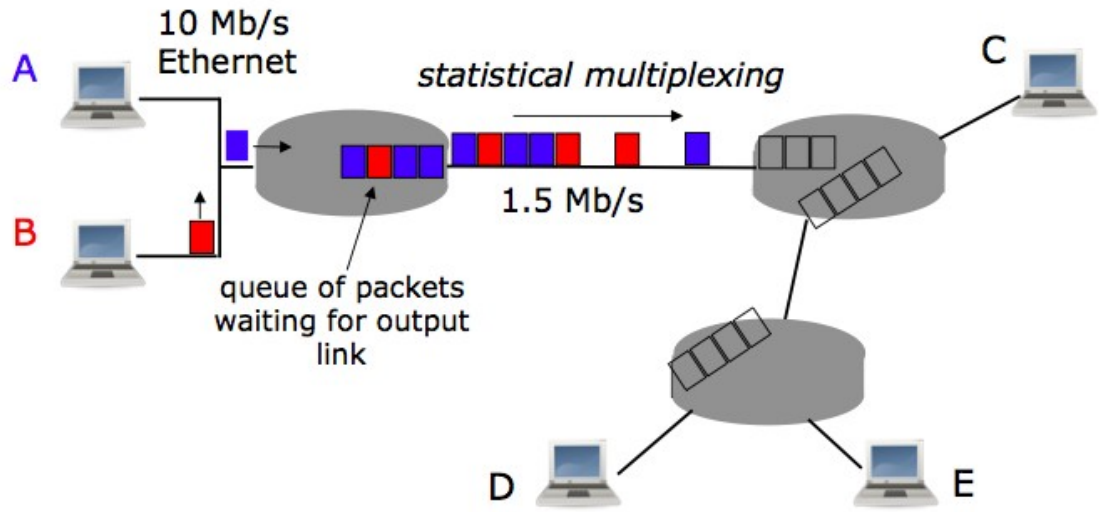
1. rete elettrica (corrente elettrica): può andare bene anche se presenta troppi disturbi;
2. rete gas (tubature del gas): la tecnologia esiste ma i tubi del gas non sono schermati per sopportare il traffico dati. In alcuni paesi la tele-lettura del gas avviene tramite le tubature a partire da un piccolo modem collegato al contatore del gas;
3. rete telefonica: l'unica ad avere un cavo di rame utile allo scopo.

La connessione dati è intesa come una questione di tempo e di tensione → teorema di Fourier (= dato un segnale lo si approssima con una mistura di seni e coseni). Con un segnale sinusoidale puro la rappresentazione risulterebbe pulita, tuttavia segnali simili non esistono così si ottengono dei *lobi* che determinano l'ampiezza del segnale. Si intende con il termine **modulazione** prendere una parte del grafico delle frequenze di un segnale e spostarne una parte al di là dell'asse delle y. L'**ADSL** trasmette in modo chiaro voce con circa 10 kbps e sfrutta la capacità di trasferimento del doppino telefonico lasciata inutilizzata dalla telefonia tradizionale.



PSTN → rete telefonica tradizionale per la voce. I modem (ADLS e non) operano una codifica di stream di bit che vanno verso e da internet. La "parte" relativa all'upstream di dati ed informazioni è più limitata per una questione tecnologica

che oggi, tuttavia, non sussiste più. Il downstream non è limitato tuttavia fisicamente risente del degrado della linea di collegamento fisica.
La banda larga è una grande menzogna → il problema è costituito dalla linea in uscita dagli armadi di rete corrispondente al lato server del modem di casa che ha una scheda di rete sulla linea della Telecom (ma quanti modem sono effettivamente allacciati a questa scheda di rete? → D-SLAM). Il livello tipico di concorrenza ADSL in Italia è altissimo: interviene qui il multiplex statistico e per sfociare in rete a partire dal modem di casa si entra in ovvia competizione con tutti gli altri modem allacciati all'armadio di rete. Le reti industriali presentano minor concorrenza anche se tecnologicamente sono la stessa cosa. Consoliamoci capendo che le applicazioni multimediali realtime non hanno bisogno di molta banda.
Nel 2001 (new economy) è scoppiata la **bolla del .com** (o della *QoS*). La banda era sempre più ampia e la connettività sempre più disponibile ed economica (con le aziende che non investivano più nel Video On Demand) → si è corsi all'approvvigionamento oltre le reali esigenze che oggi per la maggior parte costituisce risorse che giacciono inutilizzate. Dopotutto non è la banda a fare la differenza quando sono il D-SLAM e la CPU dei router di rete ad essere limitati tanto da non riuscire a gestire il traffico moderno non stando più al passo. Oggi si vuole sfruttare meglio le risorse a disposizione senza ricorrere con insistenza all'overprovisioning (che non soddisfa nemmeno tutte le casistiche di bisogno), e lo si fa via software con un discreto successo. L'unico modo per allargare la banda è attendere il ricambio generazionale.



Multiplexing statistico. Una rete a commutazione di pacchetto funziona bene perchè non sappiamo bene cosa fa. Il *multiplexing puro* detto anche MUX costituisce un dispositivo con sistema di controllo che sceglie per ogni momento quali ingressi vanno allineati con quali uscite all'interno di un router; gli host A e B non hanno un comportamento predicibile nella generazione dei pacchetti e le risorse vengono utilizzate sulla base delle richieste istantanee (shared on demand) così da predire la congestione della rete solo su base statistica. Se il sistema di controllo è semplice e si limita ad instradare di volta in volta ogni pacchetto che attraversa il router si tratta di un *multiplexing deterministico*; il *multiplexing statistico* invece può solo stimare la probabilità che un certo pacchetto superi il router e venga effettivamente mandato nella rete: in questa

maniera i pacchetti non arriveranno mai a destinazione nell'effettivo ordine in cui sono stati mandati ma può capitare che tra di loro si siano infiltrati altri pacchetti così da non garantire un'adeguata frequenza all'arrivo dei pacchetti (da qui, la *teoria delle code*). Su di una rete vera non si possono applicare le teorie delle code perchè anche aumentando la banda non risolvo il problema ma lo sposto dal canale al router.
La **banda** si divide allora in tre macro-categorie:

1. **banda nominale** → quella per la quale effettivamente si paga e che viene dichiarata sul contratto; corrisponde a ciò che l'infrastruttura di rete è in grado di farci trasmettere (se fossimo noi gli unici usufruitori). La si ottiene per intero in rarissimi casi molto fortunati. Utilizzando l'incapsulamento dei pacchetti non si arriva mai alla banda nominale poiché si tratta di una misura della banda a livello 7 mentre l'incapsulamento avviene a livello 1;
2. **banda disponibile** → informazione che indica quanti pacchetti effettivamente la rete riesce a smaltire in un certo momento (anche questa misura è operata a livello 1 nella pila dei protocolli di rete);
3. **banda garantita** → che corrisponde a tutta la banda effettivamente destinata all'uso (dove le risorse garantite sono una semplice frazione della banda nominale, trattandosi di un discorso statistico dove non c'è nulla di assicurato), ovvero le risorse che saranno sempre e comunque disponibili per il nostro traffico.

Si può intendere la banda come la capacità della rete di smaltire dati o come un range di sequenze. La larghezza di banda per il trasferimento di un bit è data dalla campana dell'istogramma del segnale. C'è differenza però tra la banda e la capacità di rete: la banda trasferisce in rete dei simboli e non dei bit (che vengono quindi accorpati nei simboli, la codifica gioca con quanti bit si possono mettere in ciascun simbolo).

Ritardo di trasmissione. Il ritardo di trasmissione è l'intervallo di tempo che intercorre tra la trasmissione di un pacchetto e la sua ricezione (riferendosi all'ultimo bit del pacchetto che risulta "invitato" solo all'invio del suo ultimo bit). Da un sacco di problemi: influisce nella modalità di fruizione del pacchetto, lo si può stimare ma non rilevare con esattezza poiché dipende dal traffico che è sempre dinamico e si può riferire, una volta stimato, ad un solo pacchetto alla volta (oltre a dipendere dal tipo di pacchetti inviati e ricevuti). Le applicazioni sono fatte per attutire questo tipo di ritardo utilizzando dei buffer che raccolgono i pacchetti che man mano arrivano dalla rete riproponendoli poi con cadenza regolare all'utente (azzerando così a livello percettivo ogni tipo di ritardo di trasmissione).

Esempio. *Un pacchetto di L bit viene trasmesso su una linea di capacità R in L/R secondi. Tutto il pacchetto dev'essere arrivato al next hop router prima di essere inviato nuovamente (per via dello store and forward).*

*Nel caso in esame: delay = 3*L/R assumendo nullo il ritardo di propagazione e tenendo conto che gli hop tra router sono 3. Se L = 7,5 Mbps ed R = 1,5 Mbps si ha che d = L/R = 5 sec ed il ritardo = 3*d = 15 sec.*

Di ritardi ce ne sono di molti tipi, lo si considera su ogni passo di un passaggio da nodo a nodo. Il **nodal delay** è il ritardo principale su ogni nodo che è composto da 4 sotto-fattori di ritardo (tra cui il ritardo di trasmissione come fattore principale di ritardo ma che deriva dal tempo necessario per smaltire i dati localmente dovuto alla scheda di rete in velocità e numero; *store and forward* ovvero quando un pacchetto per essere ritrasmesso dev'essere prima immagazzinato tutto). Si accumula ritardo per via di molti motivi: la CPU del router può essere più lenta della rete (salvo se il router è Cisco), il collegamento di uscita può essere meno capiente di quello in entrata e se la rete funziona a pieno ritmo alcuni pacchetti vengono messi in coda (in un buffer) in attesa che vengano instradati (se poi il buffer è pieno i pacchetti in arrivo non vengono immagazzinati e vanno perduti, così da dover essere spediti di nuovo accumulando ulteriore ritardo).
I 4 tipi di ritardo che compongono il *nodal delay* sono:

1. **processing delay** → detto anche ritardo di elaborazione del router, dovuto alla tecnologia della CPU del router che sceglie l'instradamento del pacchetto (solitamente di pochi microsecondi);
2. **queuing delay** → detta anche ritardo di accodamento dovuto all'attesa dei pacchetti sul link d'uscita accodati nel buffer. Se il buffer è pieno il router lavora a vuoto perchè poi il pacchetto che non trova posto nel buffer si smarrisce (dipende dal grado di congestione);
3. **transmission delay** → detto anche ritardo di trasmissione (del quale abbiamo parlato poco prima, importante per i link a bassa velocità);
4. **propagation delay** → detto anche ritardo di propagazione, dovuto al tempo che un bit impiega a propagarsi come onda elettromagnetica (ci si potrebbe chiedere allora quanto è lungo un bit. Ci sono situazioni anomale in cui un bit diventa troppo lungo per il cavo in cui è propagato e la scheda di rete può ricevere un bit sbagliato mentre è ancora trasmesso). Può passare da pochi microsecondi a centinaia di millisecondi.

Il nodal dealy è la somma di questi 4 tipi di ritardo. Un esempio di link a bassa capacità è dato dal WiFi (e sono i collegamenti che la maggior parte della gente si ostina ad utilizzare). Il ritardo di trasmissione è quello più rilevante.
A causa di questi ritardi è possibile **perdere dati e pacchetti**. Solitamente dovuti ad errori di trasmissione che oggi sono rilevanti solamente nelle reti wireless (salvo contatti del cavo di trasmissione con il cavo telefonico), causati da congestioni di buffer pieni e troppo poco capaci in situazioni di alto traffico. Le trasmissioni multimediali hanno bisogno di garanzie sulla perdita di pacchetti per essere accettabili: per questo si tende a dichiarare una probabilità massima di perdita di informazione su un periodo massimo di tempo.
Gli **errori di trasmissione** sono tipici di ogni mezzo trasmissivo: causano la perdita di pacchetti e la loro eventuale corruzione qualora arrivino. Oggi capitano molto raramente. Non si possono eliminare, tuttavia sono molto bassi: si pensi all'utilizzo dei cavi in fibra ottica. Nel caso delle reti wireless gli errori di trasmissione sono dovuti ad interferenze esterne: così il tasso di perdita non è per nulla trascurabile.
Quando c'è troppo traffico in arrivo su un nodo, i pacchetti in eccedenza alla sua capacità di elaborazione vengono accodati in un buffer: se il buffer è pieno vengono scartati. Per gestire le **congestioni** sono stati progettati protocolli a livello di trasporto per contrastare (ma non evitare) questo fenomeno. Un dato perso dovrebbe essere (teoricamente) ritrasmesso. Prima di ritrasmetterlo

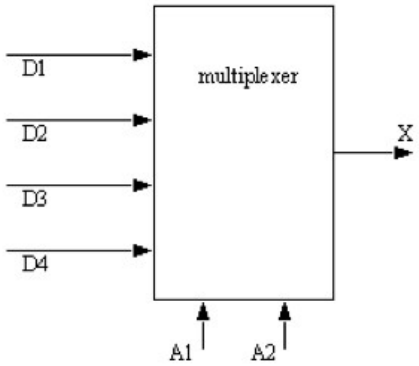
bisogna essere sicuri che il dato sia stato effettivamente smarrito: bisogna quindi attendere il *timeout* oltre a dover raddoppiare il tempo di trasmissione (motivo per cui non si usa TCP per le trasmissioni multimediali, ma UDP). Esistono meccanismi di emergenza basati su statistiche di occupazione dei router che scartano solo i pacchetti in coda da troppo tempo per far spazio ai nuovi (qui intervengono meccanismi a livello di trasporto → TCP/IP che aggiunge ritardo al ritardo e non si può scegliere se farlo intervenire o meno, salvo in caso si specifichi chiaramente di voler avvalersi di un altro protocollo di rete anche se non è detto che sia possibile scegliere il protocollo di trasporto → allora sarebbe auspicabile l'utilizzo di UDP). Si possono presentare fenomeni di F*orward Error Correction* con l'invio di pacchetti nella rete (anche ridondanti) che possano restaurare gli eventuali pacchetti successivi andati perduti (la ridondanza protegge i dati più significativi, tuttavia elimina gli effetti positivi della codifica e compressione, ma esistono delle tecniche di FEC inserite nei meccanismi di codifica). A questo punto bisogna chiedersi se è possibile permettersi un certo ritardo: dipende. Più sistemi di correzione errori si stratificano più ritardo si accumula: possono essere d'enorme aiuto oppure possono intralciare più di quanto possano tornare utili. È qui che i **sistemi di codifica** vengono in nostro soccorso: sono d'aiuto perchè sono pensati apposta per perdere dati (se impongo una certa qualità di servizio sulle caratteristiche del sistema di codifica) e non hanno un bisogno esplicito di fare delle ritrasmissioni. La codifica multimediale è studiata appositamente per sopportare la perdita di una parte di informazione.

Jitter. Misura la varianza istantanea sul tempo medio di interarrivo dei pacchetti (la trasmissione si dice isocrona se il jitter è garantito essere sempre pari a zero). Il jitter può essere solo misurato ma non lo si può controllare perchè cambia velocemente conformemente al traffico nella rete. Lo si può tamponare aggiungendo, nelle applicazioni, un buffer (come con il ritardo) ma così facendo si aggiunge un ulteriore fattore di ritardo, che è permesso in trasmissioni video ma non in quelle audio. Si può risolvere il problema in due modi: in *hardware* con soluzioni deterministiche costruendo delle reti la cui infrastruttura non ha problemi relativi alla fornitura di QoS, oppure in *software* implementando delle architetture in grado di capire se una rete IP è in grado di sostenere e soddisfare certe richieste di QoS ed in caso, aiutarla (in questa maniera non devo intervenire sui router ma si tratta di soluzioni prettamente probabilistiche → date le risorse di rete si ha un margine per l'uso delle stesse). Le **soluzioni hardware** permettono di garantire la QoS usando direttamente l'infrastruttura di rete, agevolando la separazione netta delle risorse, la protezione dei dati e la sicurezza (e sono le uniche soluzioni che in linea teorica funzionano sempre). Separando le risorse non si subisce più il multiplexing statistico a favore della *commutazione di circuito* con conseguente eliminazione della congestione ma solo se il mezzo fisico è un unico cavo, tipico della telefonia; la protezione dei dati si ha con l'eliminazione del pericolo dell'intercettazione tramite "rubinetto" nel mezzo fisico. La qualità allora è garantita a prescindere. Tuttavia l'apparato non è sofisticato ed effettua un'ammissione di chiamata per vedere se ci sono le risorse e per poter poi riservare le risorse richieste → **call admission**: ammissione al sistema che avviene quando ci si collega per inviare un flusso multimediale. Derivante dal gergo telefonico, una *call admission* è un'operazione che viene svolta quando si decide se una nuova chiamata (oppure un flusso multimediale) può essere

ammessa o meno nel sistema, dipende dalle risorse disponibili.
Bisogna capire come proteggere le informazioni: la *commutazione di circuito*, tramite apparati di rete, da la sensazione che esista davvero un circuito fisico al di sotto della comunicazione (è lo stesso concetto del circuito logico). Tuttavia non è detto che quella comunicazione sia l'unica ad usufruire del mezzo fisico: bisogna far convivere comunicazioni diverse in parallelo sullo stesso mezzo che rimangono tra loro disgiunte grazie a parametri differenti (come accade per i dati voce e per quelli di un browser sulla stessa linea) sfruttando un po' di multiplexing in base alle frequenze (come accade per i canali televisivi) ed ad altri fattori. Il traffico viene diviso a livello trasmissivo per poter continuare a garantire i parametri di rete, in pratica utilizzando la commutazione di circuito. Il problema allora si sposta su come fare a condividere lo stesso circuito da una serie di utilizzatori e la suddivisione può essere fatta:

1. sulle **frequenze** (FDM, come accade per i canali televisivi, allora si usa una banda divisa in più frequenze distinte);
2. sui **tempi** (TDM, allora si usa una banda occupata da una frequenza alla volta per un tot di tempo allo scadere del quale si passa alla frequenza successiva);
3. sulle **lunghezze d'onda** (WDM, limitata alla fibra ottica che utilizza onde luminose di diversi colori che non si disturbano tra loro);
4. sulla **codifica** (CDM, con codifica in condivisione su mutliplexing tipiche delle reti di telefonia 3G).

Quando si parla di **commutazione di circuito** si intende che per fare setup si deve innanzitutto capire qual'è la capacità delle linee e quante ce ne sono di disponibili, perchè se non ci sono linee libere non si può comunicare; si deve capire quel'è la capacità di calcolo degli apparati (la CPU potrebbe non riuscire a sostenere il calcolo); dopo la prenotazione le prestazioni sono garantite (solo a posteriori). Senza che la call admission vada a buon fine non si riesce a riservare le risorse utili per la comunicazione e questa non ha luogo. Appena però la call admission ha buon esito le risorse allocate non verranno condivise con nessun'altra comunicazione.



A1	A2	X
0	0	X=D1
0	1	X=D2
1	0	X=D3
1	1	X=D4

Il *multiplexing* ed il *demultiplexing* sono tecniche che permettono di (de)serializzare un insieme di sorgenti su un unico mezzo trasmissivo. Alcuni esempi classici del multiplexing sono il MUX elettronico ed il (de)miscelatore della tv. Si intende per **codici di linea** i metodi che su un mezzo fisico rendono possibile il multiplexing: definiscono in quale modo più sorgenti sono in grado di condividere uno stesso mezzo trasmissivo.
Codici di linea tipici sono FDM (la frequenza è divisa in parti uguali dai segnali

che la utilizzano) e TDM (frequenza utilizzata per intero da ogni segnale, ma a turni di eguale lunghezza). Sono sistemi di condivisione del canale e non esiste un obbligo di bitrate tra le sorgenti perchè presentano sistemi di codifica differenti.

Esempio. *Si immagini 4 ditte (A, B, C, D) che hanno costruito una rete stradale privata per la consegna delle loro merci. Gli automezzi delle ditte si contendono l'uso della strada ed occorre stabilire delle regole: divisione per corsia (la prima corsia la usa tutta la ditta A, la seconda tutta la ditta B, ecc) oppure divisione per tempo (in una certa fascia oraria tutta la strada è utilizzata dalla ditta A, in un'altra fascia oraria tutta la strada è della ditta B, ecc). Nelle telecomunicazioni non esistono corsie ma si parla di prenotazioni sull'uso del canale.*

Nella realtà non si usa nessuna delle due soluzioni perchè entrambe hanno problemi reali di sincronizzazione che occorrerebbe perfetta per un tempo infinitamente lungo (cosa fisicamente impossibile) oltre ad essere facilmente intercettabili dimostrando così problemi di sicurezza. Nella realtà si usa un miscuglio delle due all'interno del quale i segnali saltano da una frequenza all'altra nell'unità di tempo detto **TFM** oppure **Time and Frequency Multiplexing**, cosicché in ogni istante tutti i segnali cambino di frequenza in maniera casualmente coordinata. Questo meccanismo ha però due grossi svantaggi:

1. è difficile trovare sequenze casuali che comunque non consentano alle frequenze di accavallarsi ad ogni “salto”;
2. serve un sistema di sincronizzazione ancora più preciso di prima.

Tuttavia l'intero sistema è meno sensibile ai disturbi di frequenza (globalmente il throughput risulta essere più alto e con guadagni migliori ma con spreco di risorse e complessità alte), presenta una forte protezione contro le intercettazioni ed ha una grande capacità di trasferimento.
Le risorse rimangono allocate anche se non si stanno usando: non si dispone di più risorse ma le si gestisce e le spartisce in modo differente.

Il **CDM** o **Code Division Multiplexing** costituisce in pratica un multiplexing con codice numerico. Tutte le sorgenti possono inviare quando vogliono su tutta la banda a patto di usare il giusto “codice” per codificare i dati: l'idea è di sintonizzare le pubblicazioni in maniera matematica. Nell'etere ci stanno tutte le informazioni del mondo tuttavia con una formula matematica è possibile “sintonizzarsi” sulla frequenza giusta facendo così risaltare il segnale rispetto a tutti gli altri sparsi nell'etere. La tecnica del CDM consiste nel miscelare N flussi di bit previa moltiplicazione (=modulazione) di ciascuno di questi con una certa parola di codice C scelta fra le N parole di un codice ortogonale. Per *codice ortogonale* si intende una parola per la quale vale che tutte le parole sono ortogonali tra loro quando corrispondono a due vettori che a loro volta sono ortogonali tra loro se la loro moltiplicazione equivale a 0 nello spazio vettoriale. La parola è una sequenza di bit alla quale si associa un vettore: se il vettore ha N bit allora appartiene ad uno spazio vettoriale di N dimensioni dove allora ho N vettori indipendenti tra loro e tra loro ortogonali. Lo spazio di codifica definisce allora un autovettore, ovvero uno spazio che ha come sottoinsieme uno spazio vettoriale espresso come combinazione limitata di vettori ortogonali. I codici della codifica sono associati agli autovettori di questo spazio di codifica. Le

parole sono lunghe N simboli binari e devono avere una frequenza più alta dei semplici bit; non è un problema perchè sono gestiti dalla CPU. Le parole del codice sono costituite da N simboli binari chiamati *chip* per distinguerli dai bit d'informazione, di durata N volte inferiore al bit d'informazione.
I codici ortogonali sono derivati ed estratti da sequenze ortogonali (la cosa non è difficile) derivate da permutazioni di *matrici di Hadamart*:

matrici di Hadamart:

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}$$

$$H_{2n} = \begin{bmatrix} H_n & H_n \\ H_n & -H_n \end{bmatrix}$$

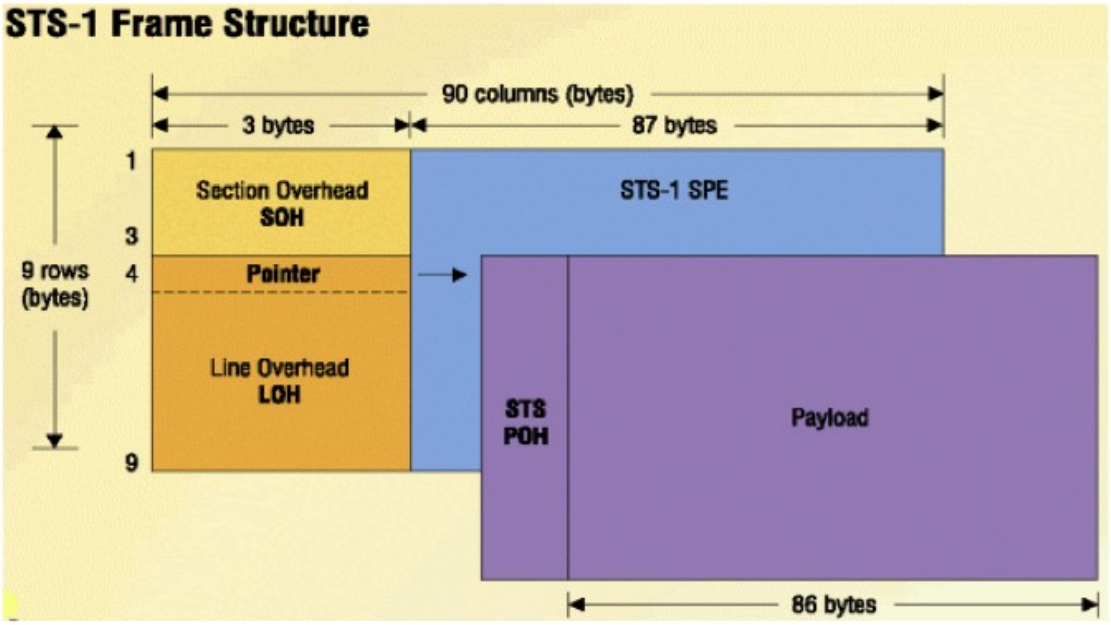
Esempio N=4

$$C_0 = \{1,1,1,1\}$$
$$C_1 = \{1,-1,1,-1\}$$
$$C_2 = \{1,1,-1,-1\}$$
$$C_3 = \{1,-1,-1,1\}$$

H_2 è la matrice degli autovettori in uno spazio bidimensionale. La matrice successiva è data dal doppio di quella precedente. Il numero di segnali che si possono miscelare è sempre e solo una potenza di 2.
Il *code multiplex* è molto comodo da usare: si sfrutta un filtro senza nessuna sincronia. Presenta grossi vantaggi in termini di efficienza, assenza di coordinamento ed una forte predisposizione alla protezione da interferenze esterne ed intercettazioni; ha anche svantaggi relativi alla sua lentezza e alla sua complessità di gestione. Ogni tanto però si è costretti a fare una rotazione nell'utilizzo dei codici per aumentare la sicurezza.

Il **WDM** o **Wave Division Multiplexing**, permette collegamenti più semplici senza aver realmente bisogno di appoggiarsi sul multiplexing: tipica delle fibre ottiche, è una suddivisione di frequenza senza multiplexing (tratta di segnali luminosi "colorati" che non si disturbano mai). La modulazione di frequenza è risolta in una maniera differente dalle precedenti generando fin da subito segnali in una data frequenza → l'informazione viene canalizzata su una certa lunghezza d'onda e questa si sposta lungo il mezzo trasmissivo.
È il meccanismo sul quale si basa **SONET/SDH** (dove SONET → *Synchronous Optical Networking* e SDH → *Synchronous Digital Hierarchy* sono due nomi diversi per chiamare due varianti minime della stessa tecnologia stabilite da ETSI ed ANSI, uno da parte europea e l'altro da parte americana). Si tratta di un sistema di trasmissione di dati su WDM utilizzando più sorgenti sincronizzate contemporaneamente → protocollo posizionato tra i livelli 1 e 2 della pila protocollare di rete, abbastanza "stupido" tanto da poter essere integrato a seconda dell'uso che se ne fa. Il protocollo permette la trasmissione dati in un circuito chiuso per l'invio di più informazioni da sorgenti sincronizzandole tra loro ed è totalmente svincolato dal protocollo utilizzato a livello di data link.
Tuttavia fa un uso di multiplexing molto pesante: questo permette alle sorgenti di avere la loro frequenza indipendente (e per allargare la banda basta cambiare

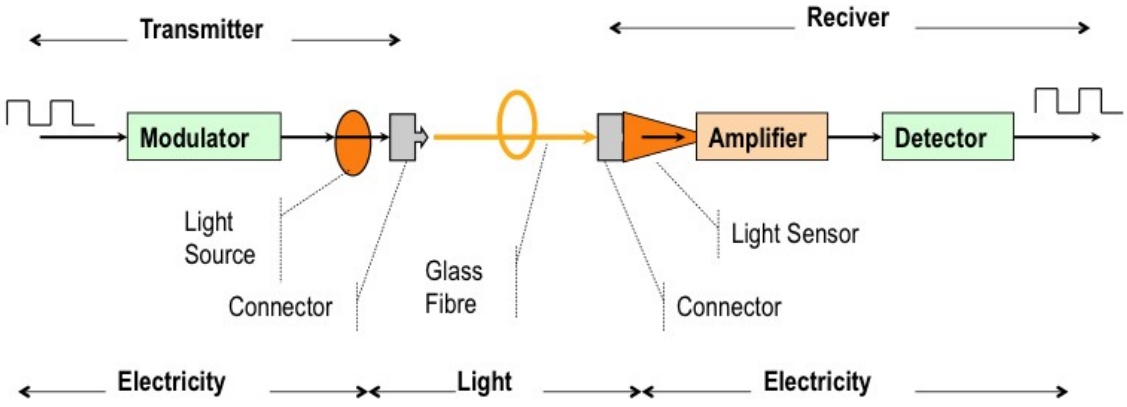
i fotodiodi ed i ricevitori alle estremità della fibra); pensate per un sistema di campionamento ad altissima frequenza; viene fatto pesante uso di *padding* per accomodare direttamente il *payload*.
Il frame SONET si chiama STM-1 (Synchronous Transport Module level 1) ed è generato con un certo rate (uno per ogni 125 millisecondi per ogni lunghezza d'onda della fibra) ma esistono anche altri STM (2, 3, ecc). Questi frame si generano sempre tant'è che alcuni possono anche essere vuoti; il ritardo di elaborazione dovuto all'elaborazione da parte della CPU (che però nelle fibre è sostituita da un prisma che opera un instradamento definito *routing ottico*) ed il ritardo di accodamento devono essere, assieme, inferiori ai 32 microsecondi. Il limite esiste (anche se difficilmente raggiungibile) per tenere sotto controllo i limiti estremi della rete a fibra (end node) dove si passa da segnale elettrico a segnale luminoso.
Il **routing ottico** è tutt'altro che banale, dovuto alla collocazione di prismi al terminare delle fibre che spezzano le lunghezze d'onda luminose instradandole. I prismi devono essere fissati ed inamovibili, sennò si rischia di spezzare la comunicazione. Viene fatto a livello di percorso, solo che per i soliti motivi di scalarità (si vedano le reti a circuito virtuale in generale) un pacchetto potrebbe dover saltare da un “colore” ad un altro (perchè i circuiti virtuali sono fatti con i colori, fissi perchè i prismi sono fissi).



Un frame SONET è costituito da una grossa matrice della quale la prima colonna è costituita da una serie di informazioni importanti (ma scarsamente sostanziose) mentre tutto il resto è dedicato al blocco per il trasporto dei dati. Il **pointer** è un trucco utilizzato da SONET per indicare a che punto del frame incomincia la sezione dedicata ai dati utili trasportati (tuttavia costituisce anche un limite perchè si possono accettare dati sino alla lettura della riga contenente il pointer, se non ne giungono sino ad allora il frame dovrà rimanere vuoto). Il payload aiuta la sincronizzazione dei dati tra diversi frame e può trovarsi a cavallo tra frame consecutivi, il chè sta ad indicare l'occupazione dell'inizio del frame successivo.
La QoS offerta da SONET offre una certa disponibilità della connettività pari al

100% della linea fissa ed una banda trasmissiva virtualmente infinita (perchè in realtà è un problema degli apparati). Tutte le tecnologie hanno un ritardo dovuto alla componente della velocità della luce, tuttavia questa fa uso appunto di luce che sfiora tale velocità annullando così il ritardo ad esso relativo: appetibile per le comunicazioni telefoniche. Il ritardo di trasmissione è il minimo ottenibile al momento perchè si parla di fibre ottiche e la probabilità di perdere dati è veramente molto bassa. Il jitter è sempre zero.

Fibre ottiche. WDM si appoggia sulle fibre ottiche. Sono mezzi trasmissivi di silicio formati da un'anima (detta *core* o *nucleo*) di silica SiO_2 avvolta in un involucro esterno (detto *cladding* o *mantello*) di silica drogata con Ge, B o Ti. La superficie tra nucleo e mantello realizza uno specchio perfetto: i materiali drogati presentano, se colpiti da certe lunghezze d'onda, una riflessione pura ed una rifrazione tendente allo zero. Produrre fibre ottiche non è molto costoso, lo è possederne (produrre 1km di fibra costa meno che produrre 1km di doppino telefonico risparmiando anche sugli apparati). La fibra presenta basse perdite di trasmissione, bassissime perdite energetiche o termiche, banda trasmissiva quasi infinita, immunità al rumore elettromagnetico, basso costo di produzione, ingombro e peso ridotti, materiale resistente e flessibile (rispetto ad altre tecnologie) e maggiore sicurezza sui dati non consentendo il *wiretapping*. Ha tuttavia problemi di giunzioni (costano molto e creano degli "scalini" nei quali l'onda luminosa potrebbe riflettere tornando indietro), raggio di curvatura oltre il quale la fibra si spezza e conversione elettro-ottica difficile, oltre a soffrire interferenze da radiazioni gamma. Quando una fibra si danneggia conviene abbandonarla per passare ad una nuova tra le tante posate ed inutilizzate.



I bit (rappresentati con l'onda quadra) passano nel modulatore che li trasforma in luce tramite un led per poi essere sparati sulla fibra: dall'altra parte la luce viene de-modulata e ritrasformata in bit. Gli impulsi luminosi delle fibre ottiche stanno fuori dalla luce visibile (luce infrarossa) ed esistono solo in tre spazi di lunghezza d'onda prestabiliti. Esistono tipi diversi di fibra:

1. fibra monomodale: usa una sola lunghezza d'onda;
2. fibra multimodale: utilizza più di una lunghezza d'onda.

La differenza tra le due è storica ed è stata imposta dall'evoluzione delle tecniche costruttive. Il numero di colori ammissibili su una fibra multimodale è limitato alle dimensioni del core. Ogni colore percorre un percorso geometrico diverso: quello che fa il percorso più lungo generalmente lo fa in un mezzo più veloce, gli indici di rifrazione sono disposti in maniera tale da ridurre le differenze

temporali. La **dispersione modale** è inerente solo alle fibre multimodali ed è causata dalla differenza di percorso dei vari colori: l'effetto è l'allargamento della durata dell'impulso da cui una limitazione sulla capacità trasmissiva massima. La **dispersione cromatica** dipende dal materiale ed è causata dalla variazione dell'indice di rifrazione con la lunghezza d'onda: l'effetto è un'attenuazione troppo forte del segnale (purtroppo non siamo capaci di produrre fibre ottiche senza alcuna impurità). L'**attenuazione** è misurata in dB/Km e rappresenta la perdita di segnale sull'unità di lunghezza: determina la massima distanza percorribile senza ripetitori.
Non si utilizzano le fibre ottiche per qualsiasi uso per una questione di costi di possesso (ok per i backbone ma non per uso domestico, per esempio con schede di rete costosissime) e perchè sono meccanicamente fragili (basta una curva neanche troppo accentuata per spezzare il cavo).

Capitolo 4
QoS Data Link

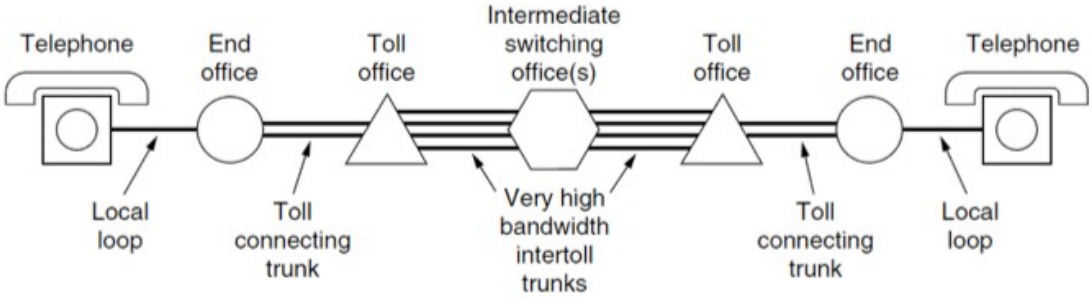
Soluzioni a livello Data Link per la qualità di servizio

Il livello **data link** nella pila protocollare ISO/OSI regola l'accesso al singolo mezzo trasmissivo fisico. Quando la maggior parte della tecnologia era su rete locale aveva senso parlare della qualità di servizio del livello data link ed aveva senso progettare protocolli efficienti che fossero in grado di gestire il concetto di qualità di servizio (per differenziare contenuti e datagrammi), ma oggi, per motivi commerciali, la tecnologia si è molto evoluta e lo studio della qualità di servizio a questo livello è diventato obsoleto, tant'è che oggi si tende anche in locale all'overprovisioning.
Il livello data link è stato molto utile a livello di qualità di servizio costituendo un protocollo per la prenotazione di *slice* sul mezzo trasmissivo (nella forma di allora oggigiorno in disuso) quando telefonia e dati viaggiavano assieme sullo stesso canale: si aveva un sistema di slot con i pari dedicati alla telefonia ed i dispari al traffico dati ethernet → si voleva però differenziare il traffico vocale da quello dei datagrammi.
Ethernet gestisce a suo modo un po' di qualità di servizio avvalendosi di certe policy per l'uscita dei pacchetti sulla rete (ma bisognava entrare all'interno della parte dati del pacchetto per capirne la tipologia e deciderne l'importanza che poi doveva servire alla scheda di rete in partenza ed in arrivo per valutarne la priorità). Sulla rete **ATM** (evoluzione naturale della rete telefonica che oggi giorno è in dismissione a favore di altre tecnologie) si possono usare una moltitudine di protocolli differenti (come il **PPP** per la comunicazione tra modem con un meccanismo di handshake riservato alla linea ADSL con lo scopo di comunicare al D-SLAM quante risorse servivano ed andavano riservate). Il **Frame Relay** (oggi dismesso) si frapponeva tra rete Internet e rete Aloha per distinguere traffico di varia priorità ma con pochissima banda a disposizione (rendendo impossibile il traffico multimediale). I protocolli **UMTS** e **MPLS** sono coloro che gestiscono il traffico dati moderno.

Ethernet. Anche sull'ethernet la qualità di servizio ha significato relativamente alla quantità di risorse allocate, solo ed esclusivamente perchè sono risorse che vanno negoziate, tuttavia in ethernet non si utilizzano più mezzi condivisi e non si ha contesa nemmeno in modalità full duplex (insomma, dipende dal contesto e può essere più o meno rilevante). Mentre un tempo si aveva anche tipologie a bus dove era vitale dare priorità a certi frame ed era fondamentale utilizzare sistemi di prenotazioni di slot temporali, oggi le reti locali ethernet sono tutte a stella con uno switch centrale e non più con un router: è lo standard costruttivo ed è d'obbligo per velocità maggiori di 100 Mbps (questo perchè ethernet si serviva di un particolare tipo di protocollo CSMA/CD, sistema di rilevamento collisioni, che ascoltava la rete rilevando onde magnetiche: l'intervallo di tempo per rilevare una collisione era dovuto tra le altre cose anche alla lunghezza del cavo fisico così da richiedere la presenza di un router ogni minimo tragitto altrimenti in caso di lunghezze superiori al dovuto non si sarebbero potute rilevare le collisioni → problema risolto con la tipologia a stella), in più in questa maniera non sono possibili collisioni tantomeno contese sul mezzo

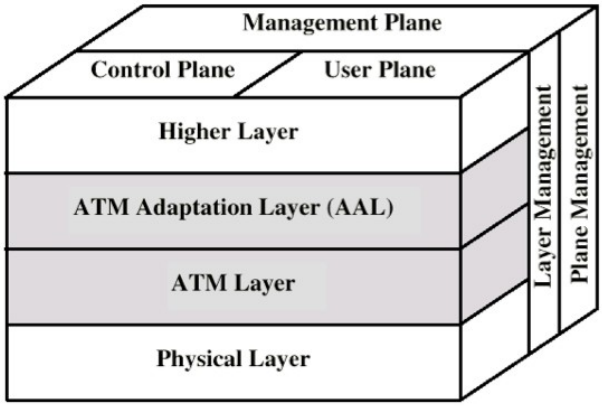
trasmissivo soprattutto se la linea è full duplex. Oggigiorno parlare della qualità di servizio su ethernet, dato le sue moderne caratteristiche, ha perso il suo senso originale.

Asynchronous Transfer Mode (ATM). Era una tecnologia di rete che doveva unificare qualsiasi trasmissione, cosa che non è successa per motivi prettamente commerciali. Si è pensato ad una rete a commutazione di pacchetto ma che non fosse come quella Ethernet perchè per le aspettative non garantiva la giusta velocità (doveva essere una rete ad altissima velocità). Anche il telefono sarebbe quindi stato a commutazione di pacchetto utilizzando pacchetti di dimensione molto ridotta detti **celle** di 53 byte, ottimi per il traffico vocale ad alta velocità. Dopotutto ATM si allacciava e faceva largo uso di tecnologie prettamente telefoniche; *Public Switched Telephone Network (PSTN)* è il nome tecnico della rete telefonica tradizionale a commutazione di circuito dove l'instradamento veniva fatto in modalità gerarchica a 3 livelli: dal telefono all'end office (armadio telefonico) considerato anche come *ultimo miglio* dalle case private di un quartiere le linee confluivano nell'armadio telefonico; dall'end office al toll office (uno per zona di interesse) le linee dagli armadi di zona finivano nel centralino telefonico; dal toll office alla centrale telefonica (una per città) si smistavano le telefonate. La rete telefonica moderna non ha però più queste caratteristiche.



Con ATM però il rinnovamento della linea ha interessato solo il "cuore" del sistema e non tutti gli apparati periferici che sono sempre gli stessi. ATM condivide con la rete telefonica alcune caratteristiche e scelte tecniche perchè in parte voleva essere il punto di partenza della linea telefonica stessa ed in parte perchè voleva far collassare voce e dati su una sola infrastruttura (e quella vocale era già molto più estesa) → si voleva convertire alla commutazione di pacchetto la vecchia rete a commutazione di circuito (ovvero lo schemino di poco prima) e si voleva poter identificare chiamato e chiamante con il numero telefonico in modo da targare ogni pacchetto con quel medesimo numero (tanto la rete non controllava i pacchetti e non richiedeva la strutturazione continua) così nel pacchetto ATM questa cosa era fatta con solo 3 byte → era l'idea della "telefonia a pacchetto" ereditando una serie di concetti di cui sopra. Bisognava conservare anche il circuito instaurato, anche solo per poter tariffare le chiamate effettuate e fare la call admission per verificare se la linea era libera in modo da riservarsi le risorse necessarie (che utilizzando i pacchetti però era sempre libera: il guaio era capire se era abbastanza libera per inviare altri pacchetti sulla linea). Così la qualità di servizio doveva essere universale per adattarsi a dati di natura diversa.
ATM divenne così una rete orientata alla connessione sulla quale venivano istituiti dei circuiti virtuali (VC come *Virtual Circuit* o *Virtual Channel*) dalla sorgente alla destinazione: il VC doveva essere istituito prima di ogni chiamata e

prima di inviare qualsiasi dato; tutte le celle di uno specifico VC percorrevano gli stessi hop poiché taluni circuiti erano costituiti permanenti ed erano inseriti manualmente nel sistema da parte di un operatore, allora i pacchetti arrivavano tutti e grazie alle call admission la cosa non risultava affatto impossibile; tutte le celle arrivano nell'identico ordine con cui sono state mandate. Tali circuiti virtuali non sono da confondere con le "reti a circuito virtuale" che sono tecniche di instradamento che agiscono a livello superiore, delle quali ATM, tra le altre cose, fa parte.
Il vero problema era allacciare ATM all'infrastruttura pre-esistente. Così si rese ATM compatibile aggiungendo alcuni layer di adattamento così da conservare un livello fisico sottostante a piacere e di qualsiasi natura e dei livelli superiori che utilizzavano tranquillamente IP mappando negli omonimi pacchetti quelli ATM.



L'**ATM Layer** implementava il protocollo di ATM, era il punto d'accesso per il livello di rete gestendo le connessioni senza bisogno di aknowledgment ed adattava le interfacce protocollari SAP di ISO/OSI (→ interfacce che nello stack protocollare ISO/OSI determinano come i dati passano da un layer all'altro). Il problema stava anche nel gestire la comunicazione tra la rete ATM ed il modello ISO/OSI così che si è dovuto mettere dei layer trasversali per gestire ATM nelle situazioni specifiche. Si noti che al livello 4 c'era SONET che faceva uso del concetto di sessioni mentre al livello 3 le sessioni scomparivano per poi essere restaurate ed utilizzare a livello 2 da ATM per il circuito virtuale (che senso aveva creare, distruggere e ricreare sempre la stessa cosa?? Le sessioni servivano oppure no??). L'ATM Layer gestiva anche la questione dei circuiti virtuali che, se perdevano pacchetti, non si adoperavano a recuperarli (nello stile delle comunicazioni telefoniche).
L'**ATM Adaptation Layer** invece era *service dependent* (dipendeva quindi dal contesto) e doveva nascondere i dettagli implementativi della rete alle applicazioni sovrastanti. Composto di due sotto-protocolli:

1. SAR frammentava e ricomponeva le informazioni di alto livello in campi per le celle e viceversa;
2. CS era un livello di convergenza dove si correggevano gli eventuali errori di trasmissione garantendo l'integrità dei dati.

Si ricorda che per *user* s'intende l'entità che s'interfaccia al SAP e non è assolutamente detto che sia umana (anzi, in questo caso sono tutti applicativi).
Il tutto per poter creare circuiti virtuali: ATM faceva uso di switch che gestivano i VC in transito a ciascuno dei quali veniva associato un livello di servizio

negoziato in fase di setup del circuito stesso (che poteva essere di diverso tipo, best effort o realtime ed altri) tramite call admission. Ogni VC aveva quindi un identificatore detto VCI (Virtual Circuit/Channel Identifier) lungo 16 bit scelti per limitare la quantità di VC creati alla volta (problema risolto comunque dalle tabelle di farwarding così da evitare il problema della scalabilità) ma che sollevavano però problemi maggiori come la difficile associazione tra ogni indirizzo ATM ed il corrispettivo IP ed il decadimento della tecnologia di routing nel momento in cui la creazione del circuito viene forzata → problemi risolti dalla creazione di tabelle di invio e di ricezione che mappavano indirizzi ATM ed IP e che venivano sincronizzate in fase di setup del circuito mentre per il routing si adottavano soluzioni già consolidate. Siccome alcuni VC erano permanenti andavano impostati a mano da un operatore.
Il VCI utilizzato al posto di un indirizzo di network semplificava i servizi di rete spostando la complessità della rete dal centro nevralgico alla periferia permettendo così la velocizzazione delle operazioni del core (per via della riduzione del tempo di calcolo sui nodi intermedi).
I parametri della **qualità di servizio di ATM** erano specificati per ogni singolo VC al quale era associato un insieme di parametri impostati sui quali si gestiva il traffico telefonico: cell loss ratio, mean cell delay, maximum tolerable cell delay, cell delay variation (e altri). Tutti parametri già visti, mentre la banda non era di fondamentale importanza per via della presenza della call admission.
ATM usava le **classi di servizio** e si ammettevano sovraccarichi quando si utilizzavano molte classi best effort; i ritardi costituivano l'eredità delle tecnologie telefoniche (anche se nelle telefonate esistono dei limiti in termini di ritardo che non possono essere ignorati). Ogni VC era obbligatoriamente associato ad una classe e le classi di servizio determinavano con quale priorità i pacchetti andavano forwardati (quindi con quale priorità il pacchetto poteva essere pescato dal buffer di coda e processato). Le classi si dividevano in realtime e non realtime, a loro volta le classi **realtime** potevano viaggiare su servizi CBR (come i flussi video che venivano generati in maniera costante prima dell'utilizzo del MPEG) o su servizi VBR (che non garantivano la frequenza basata sul bitrate per secondo); le classi **non realtime** tipiche per il loro bitrate incoerente si dividevano in VBR che garantisce un minimo di qualità (con video registrati i quali non importa che giungano con una certa regolarità ma basta che arrivino tutti senza perdere informazioni) ed in best effort (il peggio del peggio). Il VBR aveva priorità sul best effort tipico del browsing mentre nel caso realtime il CBR oggi ha senso con il traffico audio e telefonico.
Queste 4 sono le classi generali (poiché ATM ne definiva assai di più) e sono da considerarsi in struttura gerarchica. Utilizzando una politica di questo tipo si rischiava la *starvation* (→ correndo il rischio di accumulare dati mai schedulati da buttare via) mentre il best effort si contendeva la banda lasciata libera dalle altre prenotazioni.
Purtroppo ATM ha fatto una fine ingloriosa per colpa di soliti “ingordi” (tipo IBM). La sua tecnologia era molto costosa da acquisire e peggio ancora da mantenere, poco supportata dai sistemi operativi e monopolizzata da IBM che aveva fama di essere garantista verso i clienti facoltosi ed un po' meno verso tutti gli altri. Per questi ed altri motivi si è scelto poi di affidarsi ad architetture software gestite da applicazioni.
Sulle dorsali degli operatori, a partire dal 2006, si è andato cercando una semplificazione coerente: è iniziata la dismissione dei vecchi backbone ATM con sostituzione graduale. Telecom utilizzava direttamente IP su WDM ed oggi la

telefonia è cambiata arrivando direttamente a pacchetto eliminando il livello 2. in Italia la geografia aiuta richiedendo l'utilizzo di un unico backbone per tutta l'infrastruttura. La Germania aveva investito molto in una struttura ATM ad anello che è già stata smantellata. ATM è importante storicamente per testimoniare il passaggio dalla telefonia a commutazione di circuito a quella a commutazione di pacchetto.

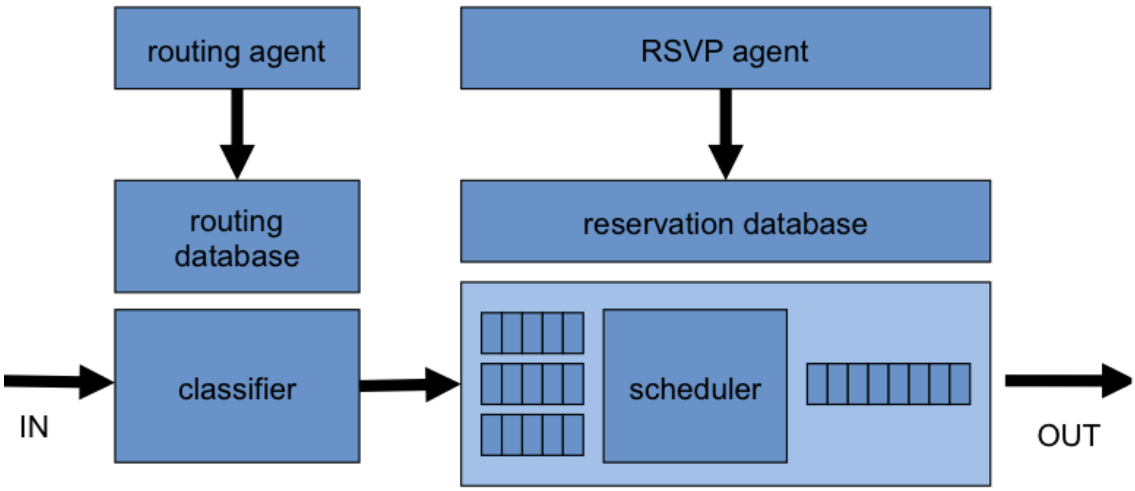
Capitolo 5
Software QoS

Architetture software per la qualità di servizio

Spesso si intende per software tutto ciò che si incontra a partire dal livello 3 della pila protocollare ISO/OSI in poi (mentre tutto quello che viene prima è sostanzialmente hardware). Tempo fa la qualità di servizio a livello 3 si divideva in 2 approcci diversificati: si lavorava con IP ma poi dall'utilizzo oculato delle risorse si è passati, per via di costi sempre più abbattuti, all'overprovisioning il chè ha dato il colpo di grazia agli approcci di un tempo in favore di altre soluzioni per l'ottimizzazione della rete. Si parla di soluzioni software perchè edificare e mantenere un'infrastruttura hardware è costoso in termini di tempo e di denaro, soprattutto è più difficile sostituire l'hardware rispetto all'upgrade di un software (anche se, in certe situazioni, anche aggiornare un software su milioni di macchine non è proprio una passeggiata) e non ci sono grossi problemi di incompatibilità tra reti differenti; se si riesce a far tutto via software si possono ottenete qualità di servizio senza obbligare gli operatori a cambiare le apparecchiature ed i protocolli. Il software a livello di rete è consapevole della forma e della funzione della rete sottostante; gli apparati fanno caso a specifici parametri per gestire il traffico di pacchetti. Grazie al software cambia il modo di gestire l'inoltro ed il traffico dei pacchetti nella rete senza intervenire sulle dinamiche di routing; si definiscono politiche di priorità per lo scheduling delle informazioni. Il software solitamente si installa *end to end* all'interno della rete lasciando totalmente inalterata la rete fisica che, al software, rimane oscura come se fosse inserita in una scatola nera, il compito delle applicazioni è infatti astrarre dalla rete sottostante.
Spuntano due soluzioni al problema, entrambe sovrastano una rete di tipo IP multiservizio (quindi formata da datagrammi che trasportano dati di natura diversa tra audio, video e browsing) senza nessun vincolo di traffico. La prima soluzione software a livello di rete prende il nome di network-aware solution nella quale si chiede agli apparati di rete di gestire il traffico in modo da garantire i parametri trasmissivi, in questa maniera non si trasmettono propriamente informazioni differenti ma gli stessi datagrammi sono associati a politiche ad-hoc; nella end-to-end solution le applicazioni “si arrangiano” indipendentemente da cosa è capace di sopportare la rete (sono soluzioni software unicamente sulle applicazioni).
Le soluzioni software a livello di rete si dividono a loro volta in integrated services (Int-serv) e differentiated service (Diff-serv) ed in entrambi i casi ci si appoggia su reti IP multiservizio: è richiesta una call admission distribuita perchè i nodi devono collaborare tra loro (se un nodo non funziona o non risponde è come se non esistesse la rete tra sè medesimo e gli altri nodi, quindi non è possibile far andare a buon fine qualsiasi call admission), si deve classificare il traffico in entrata ed in uscita (strategie di routing) ed è necessario inoltrare i pacchetti che arrivano secondo necessità dettata dalla priorità (sistema che va implementato via software). Prima si aveva a che fare con ATM in combo con SONET, poi si è passati al servizio Int-serv ereditando la logica della rete telefonica con conseguenti problemi di scalabilità, quindi ci si è affidati a Diff-serv (che ha fatto da base di partenza per un cosiddetto “ritorno al

passato") anche se oggi si preferisce lavorare con MPLS.

Integrated Service (Int-serv). Garantisce qualità di servizio ottimali ad ogni flusso distinto di pacchetti dove ogni flusso è "etichettato" da una certa qualità di servizio associata ed ogni pacchetto che appartiene ad un flusso è garantito con la qualità di servizio propria del flusso a cui appartiene. Ad ogni sessione si crea un flusso di pacchetti e si associa ad esso una certa qualità di servizio e lo si caratterizza a seconda del traffico che verrà inviato, quindi si farà distinzione tra specifiche di flusso R-spec (*reciver*, la qualità di servizio è richiesta dal ricevente) e T-spec (*transmission*, caratterizzazione del traffico che verrà inviato): si ha a che fare con ricevitori che, in veste di client, richiedono un dato servizio con certe specifiche di base (relative alla loro condizione od al contratto di cui gode il ricevitore) al server che però riceve le specifiche (molto simili a degli SLA) e ridefinisce i suoi parametri tecnologici in modo da soddisfarle il più possibile. È chi trasmette a dover poi fare una call admission per riuscire ad allocarsi le risorse necessarie per il trasferimento dei dati (anche se la richiesta è partita dal client); se la call admission fatta dal server fallisce significa che bisogna contrattare, quindi le richieste di servizio fatte dal ricevitore hanno importanza soltanto fino ad un certo punto. Serve un protocollo di segnalazione per la prenotazione delle risorse lungo il percorso.



In una architettura Int-serv si ha innanzitutto un traffico in ingresso il quale rimane nella forma di un pacchetto IP anche dopo aver attraversato il router. Un **classificatore** (*classifier*) ha il compito di stabilire a quale flusso appartiene il pacchetto di turno e per scoprirlo si appoggia alla *forwarding table* del router; in seguito si invia il pacchetto all'interno di un "meccanismo" che implementa uno **schedulatore** (*scheduler*) il quale, basandosi sulla priorità sfoggiata dalle code in ingresso, le sceglie attraverso un sistema di demultiplexer pesato. Lo schedulatore si aiuta nelle sue decisioni con un **reservation database** popolato man mano che vengono processate nuove code e nuovi flussi con informazioni di vario interesse. Alla fine di questo procedimento si ottiene il flusso di pacchetti in uscita.
Lo schema risulta scarno e semplificato perchè all'epoca dello studio di questo sistema era impensabile disporre di switch a 64 porte (ritenuti troppo costosi e troppo fuori portata) quindi non tiene conto di eventuali parallelizzazioni ed ottimizzazioni hardware dettate anche e soprattutto dalla tecnologia che evolve.

Al sistema è necessario fornire un **router agent** per controllare dall'alto i flussi che vengono immessi nel sistema e di un **RSVP agent** che implementa le call admission necessarie popolando a sua volta il reservation database (al quale comunque si interfaccia anche e soprattutto per capire se è possibile o no fare una determinata call admission).
Il server si impegna in questo procedimento basandosi su 3 modelli di servizio differenti (dove per modello di servizio s'intende una classificazione dell'insieme dei servizi di qualità di servizio supportati da una rete). Se ne possono definire a volontà ma se ne sono individuati 3 principali:

1. **Guaranteed Service** (servizio garantito) per le comunicazioni realtime con qualità di servizio ben specificata: utile per il video in diretta;
2. **Controlled Load** (caricamento controllato) per le comunicazioni con margini di tolleranza, usato solitamente per i servizi più elastici come video già registrati;
3. **Best Effort** (fare del proprio meglio), tipico dei dati di browsing.

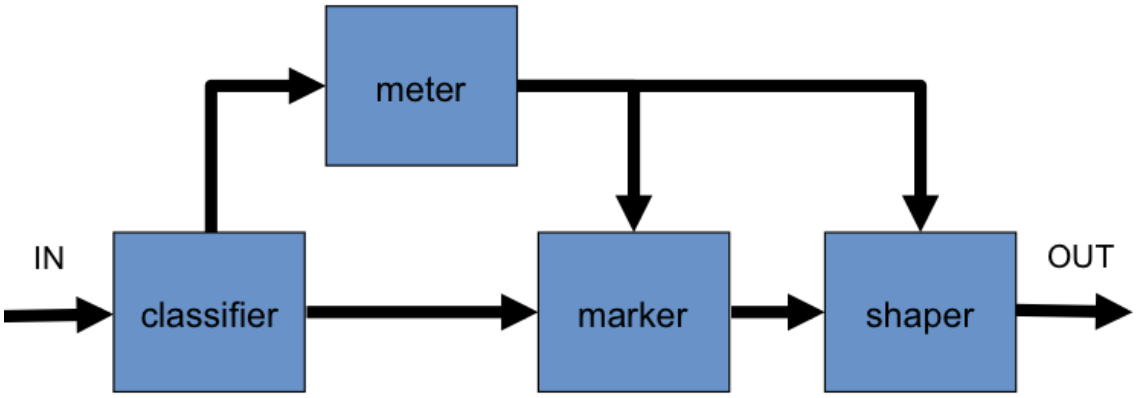
Il routing agent è un software che gestisce anche le politiche d'instradamento che però non sono più quelle standard, le quali scelgono sempre il cammino minore con il numero più esiguo di hop che, tuttavia, in questo caso non è detto sia per forza la scelta migliore (bisogna guardare alle risorse disponibili, non al tempo minore da impiegare); l'RSVP agent implementa il protocollo RSVP e si occupa principalmente delle call admission per studiare il modo di specificare il percorso più ottimale:

1. per ogni richiesta di servizio viene mandato un messaggio “Resv” contenente le R-spec contenenti le specifiche che ci si aspetta di ricevere soddisfatte dal servizio;
2. il destinatario valuta le R-spec e se accetta di inviare i dati risponde con un messaggio “Path” contenente le T-spec che sono i parametri fisici del collegamento sul quale verrà instradata la trasmissione;
3. il messaggio “Path” allora effettua la prenotazione delle risorse su ogni router attraversato in base alle T-spec.

La call admission è da ritenere andata a buon fine se il messaggio “Path” giunge a destinazione (e mentre passa su ogni router del suo cammino si assicura che ci siano le risorse adeguate per poter ritenere eseguita con successo la corrente call admission e la successiva trasmissione). Se il messaggio incontra qualche router che invece segnala di non aver le risorse richieste disponibili, prima il messaggio chiede di prendere in considerazione la seconda opzione per il cammino più conveniente e se anche questo viene ritenuto inadatto alla trasmissione il messaggio torna al router precedente chiedendo così di fare una deviazione (la quale si spera venga riassorbita durante gli hop successivi). Qualora il messaggio venisse rispedito indietro sino alla fonte significa che la call admission è fallita e non può avere luogo con quei specifici parametri di qualità di servizio, allora il server deve inizializzare una nuova richiesta rivedendo anche la qualità di servizio. Bisogna ricordare che il flusso dati viaggia in senso contrario rispetto alla richiesta e che i percorsi nelle due direzioni potrebbero essere asimmetrici. I pro di questo approccio stanno nel fatto che è di facile implementazione, tuttavia bisogna comunque modificare il software già in uso sulla rete (attendendo o meno il ricambio generazionale che nel caso del software è molto più dolce di quello hardware); potrebbero esserci comunque risorse che giacciono inutilizzate nonostante l'idea sia quella di sfruttare tutte le risorse a disposizione; la situazione non ammette scalabilità complicando la classificazione dei pacchetti a discapito del loro instradamento.

L'Int-serv è stato abbandonato per motivi economici e perchè si trattava di un protocollo scomodo da implementare sulle macchine di routing, tuttavia venne implementato su diversi apparati (tra i quali quelli CISCO) e su diversi sistemi operativi (come Windows e Linux) poiché si poteva sfruttare un vecchio computer desktop per fare routing in reti private e di piccole dimensioni.

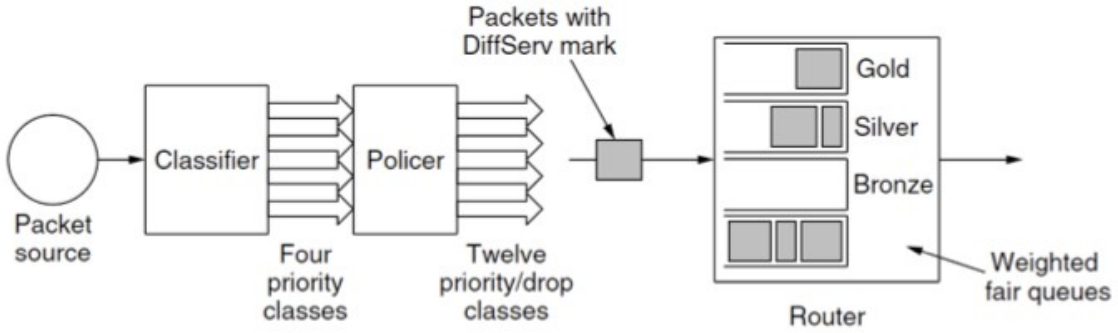
Differentiated Service (Diff-serv). Si elimina la questione del flusso e la qualità del servizio appartiene al solo pacchetto il quale entra a far parte di una *categoria*; la qualità di servizio ora non è più deterministica ma dipende se la rete agevola o meno la comunicazione così da stabilire le priorità dei pacchetti *Per Hop Behaviour* (PHB) così che la classe di servizio abbia alte probabilità di essere osservata. La rete è suddivisa in domini all'interno dei quali viene garantito un controllo Diff-serv della qualità di servizio (DS-domain); si definiscono quindi dei comportamenti hop-by-hop atti ad ottenere una certa qualità di servizio locale (i PHB di cui sopra) e la qualità di servizio richiesta diventa una caratteristica "relativa" per pacchetto e non più per flusso. Le isole sono domini di rete ristretti all'interno dei quali è definita e rispettata la stessa PHB indicata dall'amministratore di rete.



Si ha un flusso d'ingresso ed un **classificatore** (*classifier*) che si basa su alcune informazioni possedute dal pacchetto stesso per smistarlo in una data classe: si noti che la classificazione deve essere la stessa all'interno di una stessa isola ed è dettata dal PHB, basta che siano i router di confine a farla visto che una volta all'interno dell'isola non sarebbe più necessaria; una volta smistati i pacchetti sono etichettati da un **marcatore** (*marker*) che scrive un codice all'interno del loro header IP (si noti che all'interno di ogni header IP ci sono proprio 3 bit riservati a tale scopo) dando la possibilità di smistare i pacchetti in $2^3 = 8$ classi differenti. Dopo l'etichettatura il pacchetto può viaggiare tranquillamente nella rete, tuttavia passa attraverso uno **shaper** che "screma" il traffico in eccesso classificandolo come traffico di tipo best effort. Per contabilizzare il traffico si utilizza un **meter** che aggiorna i contatori utilizzati da marker e shaper per etichettare e scremare i pacchetti.
Poiché questo meccanismo è assai più raffinato e complicato del precedente, è possibile condizionare il traffico facendo admission controll solo all'ingresso dell'isola così da avere un servizio di tipo end-to-end. Occorre stabilire dei sistemi di contrattazione sulla qualità del servizio offerta nel transito tra isole perchè ogni operatore è autorizzato a definire la qualità di servizio secondo suoi criteri interni, per questo ci vengono in aiuto le classi di servizi indipendenti

dalla rete. Per poter permettere ad operatori diversi di avere un terreno comune di confronto nella contrattazione tra isole adiacenti, IETF ha introdotto delle **classi di servizio indipendenti dalla rete** (o, se vogliamo, buone per tutte le occasioni):

1. **Expedited Forwarding**, che si suddivide in 2 sottoclassi di traffico normale e di traffico con priorità (il quale dev'essere in grado di attraversare la rete come se non esistesse traffico di altro tipo). Di fatto, si prelevano dal buffer di un router prima i pacchetti marcati come prioritari e si fanno passare anche gli altri solo se non ce ne sono di prioritari in attesa (il chè però può causare la *starvation* dei pacchetti non prioritari nel peggiore dei casi);
2. **Assured Forwarding**, che divide il traffico in 4 sottoclassi ad ognuna delle quali vengono allocate delle proprie risorse (gold, silver, bronze e best effort). Inoltre, per ogni sottoclasse esistono 3 priorità di scarto per i pacchetti soggetti a congestione così da determinare 12 casistiche differenti per classi di servizio distinte. Il pacchetto viene classificato in una delle 4 classi e viene accodato alla relativa coda la quale viene schedulata a seconda del traffico che generano. Dalla sua ha che è il metodo meno invasivo per le apparecchiature, è più scalabile e le analisi simulative prospettano rendimenti migliori, tuttavia attualmente non esistono implementazioni commerciali.



Diff-serv è meno invasivo sulle apparecchiature perchè sui router bene o male esistono già meccanismi supportati per il routing gerarchico e perchè una rete solitamente è formata da hardware tutto dello stesso fornitore, inoltre solo i router di confine sono interessati a dare una priorità ai pacchetti lasciando che la rete interna all'isola (intesa anche come *autonomous system* all'interno dei quali i router condividono gli stessi protocolli) in pratica se ne disinteressi; è più scalabile perchè i calcoli su ogni pacchetto diminuiscono come rimpiccioliscono le tabelle di routing; il rendimento della rete è più alto a livello di traffico. Di contro si tratta però di una tecnologia che nessuno ha mai implementato a livello commerciale (salvo in rarissimi casi ed in maniera molto embrionale) dovuto al fatto che esiste un punto debole che non è mai stato corretto (per impossibilità ma soprattutto per perdita d'interesse): manca la call admission distribuita che è difficilissima da implementare. Ci si è chiesto se ne valeva davvero la pena ed alla fine si è preferito abbandonare il progetto.

Capitolo 6
MPLS

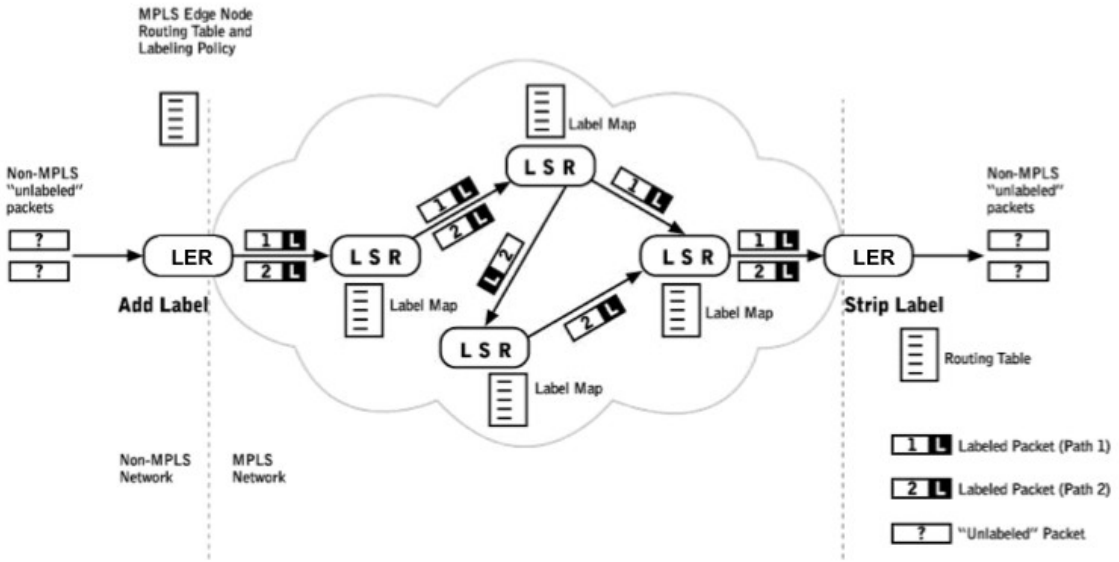
Multi Protocol Label Switching

MPLS è il protocollo che usano praticamente tutte le compagnie telefoniche moderne; è una specie di ammissione di colpa (prof. cit). Nasce per risolvere due problemi: l'isolamento del traffico per motivi puramente commerciali e per rendere isocrona la rete sottostante (dove per *isocrona* s'intende *sincronizzata*). Come protocollo non si ha ben chiaro a che livello dello stack ISO/OSI piazzarlo anche se alla fine si è convenuto di posizionarlo idealmente tra il secondo ed il terzo livello (per motivi che si discuteranno poi); pochi altri protocolli hanno il coraggio e le caratteristiche per piazzarsi a cavallo di due livelli diversi.
MPLS è una strategia per attuare la commutazione di pacchetto su reti ad altissime prestazioni (l'idea era fondamentalmente di rendere facile creare link virtuali tra nodi collegati in maniera diretta); per ottenere ciò si è riciclata una vecchia tecnologia che, apportando certe migliorie, si è rivelata vincente: ATM che facilitava di molto la commutazione di pacchetti. MPLS come ATM prima di lui utilizza un tot di bit come etichetta per marchiare i propri pacchetti (sulla cui etichetta è basato tutto il meccanismo di forwarding del pacchetto), l'unico modo per sfruttare al meglio l'hardware in fibra a disposizione creando così dei link virtuali tra schede di rete tra le quali si estende spesso una rete molto complessa ed articolata (dando vita così al concetto di *Virtual LAN* → VLAN); in questa maniera si risolvono anche problemi di sicurezza, di broadcast e di telefonia se il link venutosi a creare è adeguatamente protetto. Attraverso MPLS è possibile sfruttare qualsiasi protocollo, tuttavia non possiede un sistema di routing suo personale perchè si appoggia sul livello 3 ed incapsula i datagrammi IP (come avviene con il datagramma IP, MPLS può incapsulare qualsiasi altro tipo di dato e di pacchetto appartenente a qualsiasi protocollo di rete). In questa maniera è possibile fare davvero telefonia e se le risorse lo consentono si può ottenere anche l'isocronia.
I pregi di questo protocollo si basano sulla sua indipendenza dal livello di data link (anche se visto il suo utilizzo varrebbe la pena di adattarlo al protocollo IP); è estremamente scalabile e permette il controllo del traffico sul percorso (sia chiaro che non si crea un link virtuale ad ogni telefonata, quindi ad ogni utenza, ma ad ogni firma di un contratto, così da instaurare il link e da non doverlo più modificare per lungo tempo per ridurre così la probabilità di combinare qualche disastro) → traffic engineering; ogni tipologia di traffico capiti può essere canalizzata indipendentemente dal sistema di trasporto sottostante e permette l'*internet working* tra reti tecnologicamente differenti.
L'idea di MPLS è nata dalle ceneri della tecnologia ATM che principalmente aveva fallito l'integrazione con il protocollo IP nonostante i vari (e vani) tentativi di rendersi compatibile con ciò che era già esistente, tuttavia l'idea di fondo della commutazione di pacchetto era vincente (peccato che un datagramma IP fosse troppo grande per stare in una cella ATM, così si venivano a creare delle processioni di celle ATM con un solo datagramma IP al loro interno che potevano smarrirsi a pezzetti rendendo così l'invio del datagramma non valido con perdita di tempo e di risorse non indifferente). Ai tempi di ATM la commutazione di pacchetti grandi e di dimensioni variabili si riusciva a fare solo

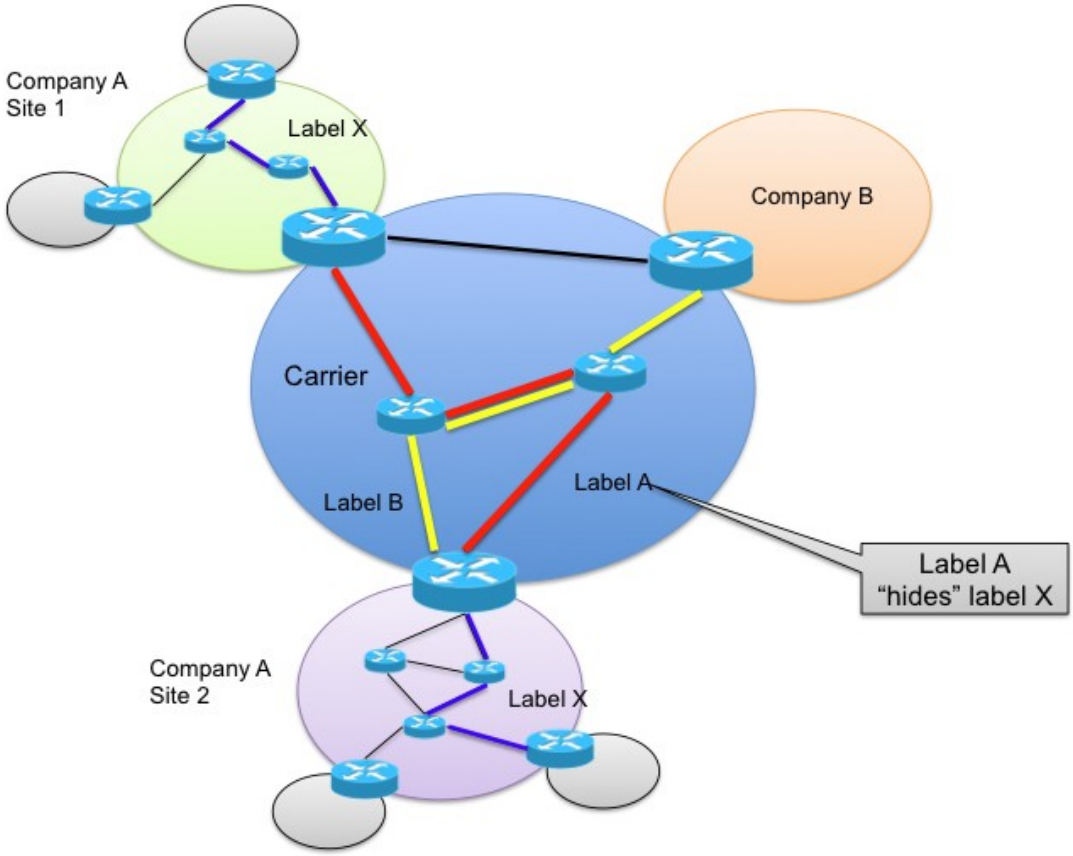
via software; MPLS ha creato allora celle più grandi di quelle di ATM all'interno delle quali non solo IP sta comodamente, ma anche qualsiasi altro tipo di trasmissione e di pacchetto; in più c'è spazio anche per l'etichetta o per l'eventuale stack di etichette apportato al pacchetto. Ai tempi di ATM la cosa fallì perchè non si era tecnologicamente pronti; MPLS è l'unica tecnologia che ha goduto di questa occasione con la comparsa di tecnologie di switching a livello 3 che non si allineavano attraverso i MAC address ma con gli indirizzi IP per creare LAN virtuali.
MPLS si trova quindi a cavallo tra il livello 2 ed il livello 3 dello stack ISO/OSI (ad una sorta di livello 2.5) poiché possiede il concetto di pacchetto ed utilizza i servizi del livello di data link, così da non poterlo effettivamente piazzare né a livello 2 tanto meno a livello 3 poiché fornisce servizi a qualsiasi livello di rete e non possiede il concetto di routing. Cosa significa? Che il modello ISO/OSI andrebbe rivisto e svecchiato. Bisogna distinguere tra architettura e modello, dove il modello è una formalizzazione del comportamento di trasferimento delle informazioni ed indica quali comportamenti debbono assolvere i vari livelli dello stack, mentre l'architettura è un'istanza del modello che implementa i vari protocolli.
Si può vedere MPLS come evoluzione di ATM poiché è stato definito tenendo presente le principali problematiche che hanno reso ATM un buco nell'acqua: si è definito meno overhead; si è stati in grado di offrire un servizio a circuito virtuale anche con frame e pacchetti di lunghezza variabile (quando le celle erano di lunghezza fissata); si è operata una migliore gestione dei pacchetti di grandi dimensioni (perchè nelle reti moderne non si può continuare a parlare di pacchetti microscopici). Dove ATM è già stato soppiantato MPLS offre una tecnologia di convergenza per offrire connettività su più larga scala.

Funzionamento di MPLS. Ogni **pacchetto MPLS**, di qualunque natura esso si riveli, viene introdotto sulla rete da una sequenza di etichette (caso nel quale i pacchetti sono incapsulati in una serie di pacchetti MPLS ciascuno dei quali avrà la sua etichetta in testa, si parla allora di stack delle etichette o label stack).
Ogni etichetta del label stack consta di 20 bit che si occupano di identificare il tipo di traffico di appartenenza, 3 bit per definire la classe di traffico, 1 byte per indicare il TTL (time to live → contatore del massimo numero di hop che può affrontare il pacchetto prima di venir naturalmente eliminato) del pacchetto ed un flag che indica qual'è l'ultima etichetta dello stack, per un totale di 32 bit uguali a 4 byte di informazione relativa all'etichetta. La commutazione a livello di router è effettuata basandosi sull'etichetta in cima allo stack (che durante il viaggio più essere modificata oppure eliminata all'uscita dell'area di competenza, in questo caso subentrerà l'etichetta subito successiva all'interno del label stack).
Le **reti MPLS** sono costituite da due tipi differenti di router: i **Label Edge Router** (LER) che costituiscono i router di bordo dell'area di competenza della rete e che classificano il traffico in ingresso ed in uscita dall'area (sono loro ad etichettare i pacchetti ed a rimuovere da essi le etichette già esistenti), questi router ignorano come sia fatta la rete interna all'area tuttavia possiedono tabelle di routing grazie alle quali scelgono quale etichetta dare a quale pacchetto in transito; i **Label Switch Router** (LSR) che sono tutti i router interni alla rete possiedono ciascuno una tabella di forwarding per effettuare l'istradamento dei pacchetti all'interno della quale si associano schede di rete in entrata ed in

uscita a seconda dell'etichetta di ogni pacchetto (la cosa non elimina del tutto le congestioni).



Si può parlare anche di *push e pop delle etichette* (dopotutto, le si considera impilate come in uno stack). I LER si dividono in router d'ingresso, i quali fanno push di etichette, e router d'uscita che si occupano di fare pop delle stesse.
Ma cosa succede quando si crea una gerarchia di domini MPLS?



Esempio. *Si immagini una rete MPLS con 3 router di confine; il carier prende*

in consegna la richiesta dell'azienda A la quale vuole instaurare un link tra due sue sedi distanti ed associa al contratto un'etichetta A (la quale fa in modo di trasformare la rete sottostante in una specie di unica LAN). Arriva poi una seconda azienda B che chiede anch'essa al carier di fare un contratto per una rete che viene rilasciata con una etichetta B. Le due aziende potrebbero essere a loro volta carrier di altre aziende e fornire ad altri clienti lo stesso servizio che loro stesse hanno richiesto, creando così una sottorete gerarchica MPLS → in questa maniera un pacchetto che deve viaggiare attraverso la rete verrà etichettato uno volta e poi incapsulato in un nuovo pacchetto con una nuova etichetta: i link A e B incapsuleranno e nasconderanno i pacchetti provenienti dalle reti X. In questa maniera aumenta la scalabilità del sistema.

I pacchetti che giungono nella rete privi di etichetta vengono marchiati dai router di confine i quali possono imporre un'etichetta alla volta (oppure più di una a seconda dei casi): gli ingress router prima di etichettare però devono identificare una Equivalent Forwarding Class (EFC), quindi forwardano il pacchetto. L'egress router all'uscita della rete invece toglierà tutte le etichette rendendo il transito nella rete MPLS invisibile alle applicazioni (può succedere però che si cada in errore e si tolga da un pacchetto non etichettato un'etichetta che non esiste, così verrà privato di 4 bit dell'header IP danneggiandolo in maniera irreparabile). Si ricordi che un router può essere di entrambe le tipologie poiché il flusso dei dati e bidirezionale.

Virtual Private Network. La sua creazione è naturale se si ha a che fare con link di natura virtuale. VPN gestisce del traffico protetto e simula una rete LAN all'interno di una rete più complessa a livello 3: in questa maniera si ha anche una garanzia sulle prestazioni e sulle risorse. MPLS permette di definire delle VPN per l'erogazione di servizi tipici delle LAN (ad esempio DHCP) anche su lunghe distanze e la protezione dei dati, invece, dev'essere gestita dal protocollo di trasferimento perchè MPLS non sa come fare la cifratura ed è indesiderabile che se ne occupino direttamente i router (rischiando di appesantire troppo i processi di instradamento).
Il **Label Distribution Protocol** (LPD) è un protocollo grazie al quale gli LSR si scambiano etichette ed informazioni sulla raggiungibilità (si tratta di un protocollo di segnalazione che fornisce un servizio di coordinamento). Questo protocollo controlla anche la presenza di eventuali errori che potrebbero creare dei loop indesiderati attraverso la creazione di **Label Switch Path** (LSP) per la distribuzione del traffico (l'equivalente di PVC di ATM). Un LSP riporta anche il percorso di ciascun pacchetto nella rete relativo ad una certa etichetta. Gli LDP sono generalmente la trasposizione di protocolli di segnalazione già in uso per le reti a circuito virtuale.
Le etichette sono assegnate con dei criteri precisi consultando innanzitutto le tabelle dei router d'ingresso:

1. **Topology Driven (TAG)**, tipico del routing classico dove le etichette sono pre-assegnate relativamente ad un certo percorso e non esiste alcun tipo di latenza al momento dell'instradamento (perchè il link è già stato instaurato). È un meccanismo che per natura si può forzare indicando manualmente quali etichette associare a quali link, ma la cosa crea casini quando poi un link viene a mancare e non è possibile riassegnare dinamicamente l'etichetta così tutti i pacchetti eventualmente etichettati con quell'etichetta andranno quindi perduti;

2. **Request Driven (RSVP)**, tipico di quando non si ha a disposizione una connessione sempre attiva, avviene su richiesta alla creazione di una connessione e potrebbe richiedere l'approvvigionamento di un grandissimo numero di etichette. Può capitare che viaggiando nella rete vengano assegnate etichette duplicate che poi andranno cambiate in corso d'opera aggiungendo ritardo sull'instradamento del pacchetto (specialmente quando ho dei percorsi molto lunghi sui quali rischio di cambiare moltissime etichette);
3. **Traffic Driven (Ipsilon)**, la creazione e l'assegnazione di una etichetta scatta nel momento in cui il pacchetto arriva al router di bordo, quindi creo per ogni pacchetto una rete ad-hoc (la cosa poi da luogo ad una certa latenza nell'instradamento perchè il pacchetto dovrà attendere l'assegnamento dell'etichetta). Utile ed apprezzabile per le comunicazioni telefoniche, questo è il metodo meno utilizzato.

Tutti e tre questi metodi rischiano di dare luogo a dei cicli che possono essere infiniti. Tali cicli possono essere gestiti da metodi particolari come l'utilizzo di protocolli di routing con approcci distribuiti (tipo *distance vector*) che però risultano un po' troppo lenti a livello di "interrogazione" della rete rischiando di costruire un percorso sbagliato, basato su informazioni non ancora giunte. A meno di accorgimenti particolari si utilizzano altre tecniche per evitare la formazione di cicli:

1. **Loop Survival**: si sfrutta il campo del TTL (time to live) allo scadere del quale il pacchetto viene scartato ovunque esso si trovi, in questa maniera se si incastra in un loop prima o poi deperirà da solo, risolvendo il loop. Si può gestire il loop anche attraverso lo sviluppo di un protocollo che converge molto velocemente oppure implementando il *fair queuing* grazie al quale le code in entrata nella scheda di rete dei router più lunghe perdono priorità a vantaggio di quelle più corte, poiché se la coda si allunga troppo significa che si sta verificando un ciclo (così si penalizza la coda-ciclo togliendole man mano le risorse di cui necessita sino alla sua eliminazione fisica);
2. **Loop Detection**: anziché arginare subito i danni da loop si preferisce attendere che la presenza di uno o più cicli venga segnalata, allora si agisce di conseguenza; esistono metodi basati sulla teoria dei grafi per l'individuazione dei cicli oppure in alternativa si manda un messaggio "in esplorazione" solitamente del tipo **LDCP** (*Loop Detection Control Packet*), il quale esplora il path del pacchetto sino alla fonte e se incontra un nodo due volte ne manda una segnalazione al punto di partenza. Se il ciclo viene verificato si elimina l'etichetta che lo ha creato interrompendo il path e riprendendo poi tutto d'accapo: i cicli eliminati devono essere sostituiti facendo routing a livello di rete;
3. **Loop Prevention**: in questo caso i cicli sono fisicamente prevenuti e facilmente evitabili, si attua una procedura di prenotazione dell'etichetta durante la quale si controlla che non esistano cicli nella rete e che il percorso sia effettivamente aperto; l'etichetta viene confermata solo in assenza di loop in atto. L'idea funziona solo se si ha parecchio tempo da perdere (cosa inaccettabile nelle comunicazioni telefoniche). In questa casistica si associa un router in entrata nell'isola ad ogni possibile router di uscita.

La qualità di servizio passa un attimo in secondo piano perchè si hanno 3 bit per l'etichetta (i quali costituiscono il QoS del servizio in questione!), quindi per

segnalare la priorità del pacchetto nella rete: sembrano pochi ma bastano per quello che si deve fare quando accade che si sfrutti una piccola implementazione di diff-serv. In questa maniera si vuole abbassare il jitter al minimo valore possibile, perchè quasi tutte le reti classificano il traffico secondo un modello "olimpico".

Capitolo 7
End-to-end

Soluzioni software end-to-end

Si parla delle soluzioni software per garantire la qualità di servizio in un sistema end-to-end. Fino ad ora si è dato per scontato di poter decidere tutto sulla struttura della rete (quali sono i nodi, come interconnetterli, come partizionare la rete affinché funzioni al meglio), tuttavia un fornitore di servizio come lo è Youtube non conosce assolutamente la conformazione della rete sottostante. Si è scelto di dedicarsi a **soluzioni software** per una questione di comodità e convenienza (soprattutto perchè sono le uniche implementabili, le applicazioni alla fine tendono ad arrangiarsi con quel che hanno sottomano); tutto questo viene chiamato ostinatamente *streaming* ma non è esatto perchè lo **streaming** è una modalità di fruizione di dati virtualmente infiniti e non sempre si associa al concetto di multimedialità; costituisce un flusso continuo di informazioni associato ad una quantità di banda. Oggi il termine streaming è utilizzato per identificare una quantità variegata di casi, tant'è che rientra nella definizione anche una pagina web (nonostante i suoi dati non siano assolutamente infiniti). Come sostrato si utilizzano sempre reti multiservizio che per definizione sono utilizzate da tantissimi altri servizi di natura non omogenea (multimedia, web, ftp, p2p) a meno di una rete costruita apposta per la multimedialità, ma è assolutamente antieconomica, così consta di una quantità imprecisata di traffico trasversale. Se la rete sostenesse un traffico prettamente omogeneo si potrebbe pensare di adottare politiche basate sul diff-serv oppure sull'int-serv, tuttavia sono assunzioni che non si possono mai fare. Si studiano allora tecniche che non riguardano più la natura degli apparati per fornire un servizio “decente” (ma cosa può essere definito “decente”?) nonostante quello che succede sulla rete.

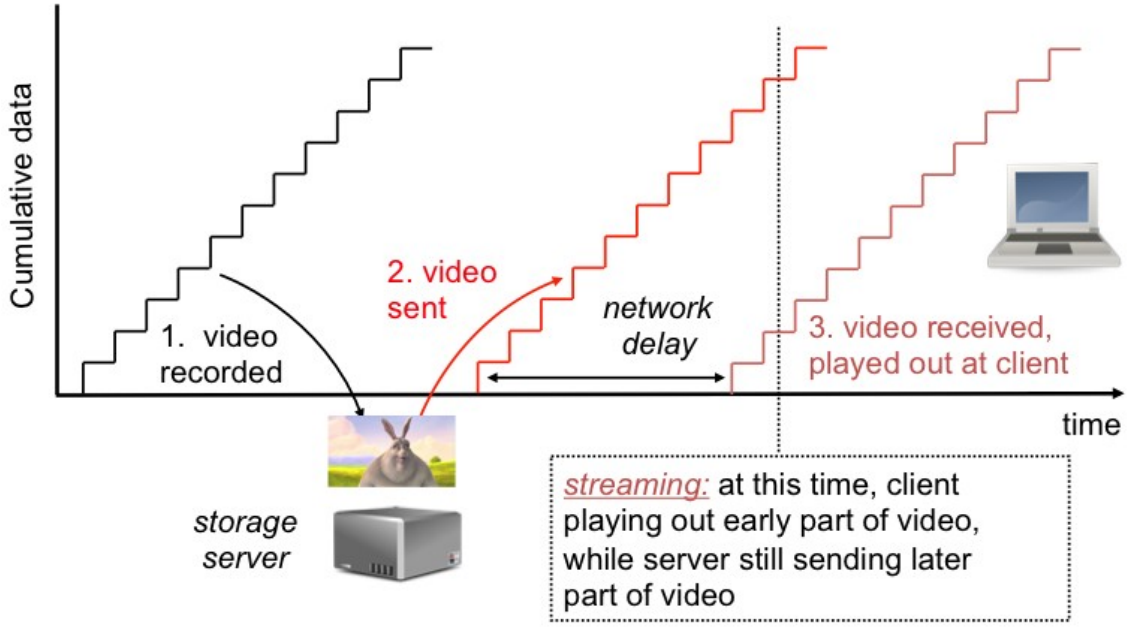
Esempio. *Misurare ciò che accade sulla rete non è affatto banale, si pensi ai tool che misurano la quantità di banda di una rete domestica. Alla fine quanto è significativo il risultato riportato dal tool? Lo è poco, perchè per esempio ignoro qual'è il traffico trasversale e non conosco affatto le risorse della rete le quali potrebbero persino migliorare il mio valore di banda (oppure limitarlo, chissà); se per banda s'intende la capacità della rete è difficile capire come eseguire la rilevazione (spesso s'innesca una successione di invii di coppie di pacchetti a distanza fissata la cui spedizione aumenta in frequenza con il passare del tempo; a destinazione verrà quindi fatto un grafico dei tempi e delle prestazioni relative alla banda d'invio e di ricezione dei pacchetti campione, che sono tutti della medesima grandezza, individuando nello schema delle aree di concentrazione dei dati. Una di quelle aree indicherà il valore della banda, ma per determinarla ci si basa su procedimenti matematici sofisticati ed accurati, rendendo questo processo lungo, complicato ed oneroso).*

Per livello di “decenza” si intende una definizione di qualità di servizio che deve essere garantita e data a priori che, anche nel peggiore dei casi, non dovrebbe mai essere oltrepassata. In questo caso, tuttavia, non è possibile tradurla in vincoli tecnologici non avendo il controllo dell'infrastruttura sottostante, così da

individuarne solo il livello minimo. Tentare di alzare la qualità di servizio risulta essere una scelta insensata.
Lo **streaming** è nato come moda degli anni '90 quando la qualità di servizio era sottoposta ad un controllo molto più rigido di quanto si faccia oggi: la sua definizione prevedeva un contenuto multimediale in rete che doveva essere mandato e ricevuto secondo un profilo controllato. Siccome questa definizione potrebbe risultare troppo larga, oggi si crede che qualsiasi cosa circoli sulla rete sia in *streaming:* in realtà il discriminante è dato da come i dati vengono prodotti, in maniera più o meno continua. Esso costituisce una successione data da un algoritmo (che non si può e non si vuole fermare) che determina un flusso di dati potenzialmente infinito e che non si può mai troncare. Dovrebbe essere inteso come streaming qualsiasi contenuto fruibile prima ancora che sia stato scaricato del tutto oppure un contenuto inviato in maniera asincrona ed in continuità con costanza ogni 25esimo di secondo (la sorgente di un contenuto non può saltare l'invio di un pacchetto mentre il destinatario può permettersi di perderne qualcuno) e che potrebbe essere virtualmente infinito. Il fatto che lo streaming sia sfruttato per i contenuti multimediali è solo un valore aggiunto.
La definizione di streaming può essere applicata a due tipi di contenuti:

1. *contenuti pre-registrati* (stored content, stored multimedia);
2. *contenuti generati sul momento* (real-time multimedia)

Stored Multimedia. Si veda il grafico seguente.



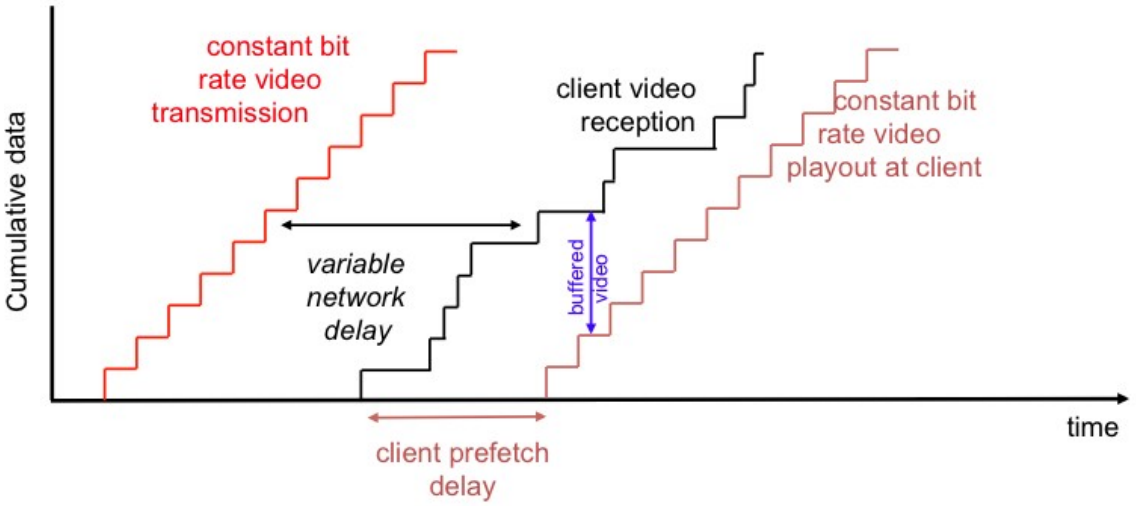
Sull'asse delle x è riportato il fattore tempo, mentre su quello delle y i dati visti in maniera cumulativa. Nella realtà lo schema a gradini regolari non solo non è ottenibile, ma proprio non esiste: in questo caso si tratta di un'approssimazione. S'invia un pacchetto (di dimensioni standard) alla volta in un intervallo regolare di tempo così da caricare il contenuto multimediale (nel nostro caso un video) all'interno di uno *storage*. Quando avviene una richiesta di fruizione da parte di un client, il contenuto viene estratto dallo storage e viene mandato al browser in streaming sulla rete, quindi verrà visualizzato dal client. Tutto ciò accade in un mondo ideale, poiché il ritardo di palinsesto potrebbe essere molto lungo come anche molto corto, a seconda dei casi, e non è assicurato che il primo pacchetto

dello scaricamento parta quando la registrazione sia già stata completata. Il quantitativo di risorse impegnate sulla rete dipende dall'invio e dal consumo del contenuto perchè si tratta sempre di byte in viaggio che comunque impegnano risorse di rete. È importante avere, da qualche parte della rete, delle capacità di immagazzinamento dei dati (come server dedicati).
Solitamente il contenuto viene prodotto e compresso prima del caricamento sulla rete quindi si possono conoscere durata, dimensione, profilo di traffico (quindi come impegna la rete) ed eventualmente si possono avere più formati a seconda delle evenienze; tuttavia con i prodotti real-time non si possono avere certezze, tutto ciò che si genera viene preso e mandato nella rete così com'è. Il vantaggio sta nel fatto che così il contenuto lo si può "andare a prendere".
Lo streaming è di tipo *pull*. L'accesso ai contenuti registrati avviene in 3 modi:

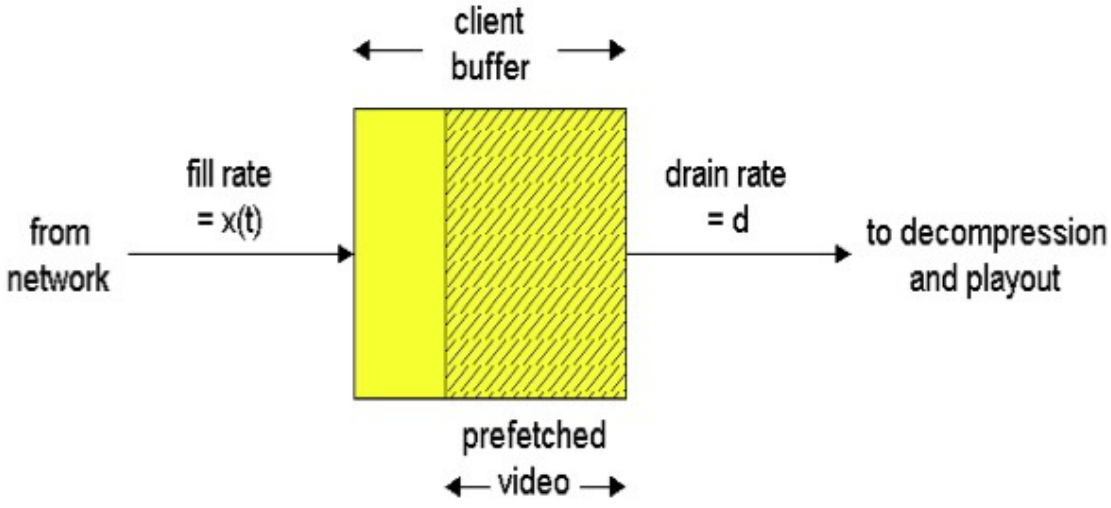
1. **Modo 1.** Esistono dei server ed i file su questi server sono tutti oggetti HTTP. Una volta che i dati sono stati scaricati dal server parte subito la loro visualizzazione. Non esiste la pipeline e bisogna aspettare a lungo prima di poter fruire interamente del contenuto scaricato. In questa maniera non si può parlare di streaming perchè la fruizione del contenuto avviene solo dopo il suo completo scaricamento.
2. **Modo 2.** Con un approccio più complicato il browser si rivolge al server il quale avvia il download non del contenuto richiesto ma di un suo metadato che lo descrive in tutte le sue parti. Sarà poi compito di un lettore multimediale interpretare il metadato e scaricare il vero contenuto in modo tale da poterlo visualizzare. Il metadato in formato XML era tipico di *Quicktime*; in questa maniera tramite HTTP arrivavano byte di conenuto già fruibili.
3. **Modo 3.** Tipico di *Media Player*, il browser recuperava il metadato da un comune webserver e lo passava al riproduttore multimediale il quale recuperava le informazioni del contenuto all'interno di uno streaming server, spesso proprietario e chiuso. In questa maniera è possibile utilizzare protocolli differenti da HTTP (ad esempio, UDP) sfruttando la banda per diagnosticare i pacchetti persi. Il software proprietario ed i protocolli proprietari, tuttavia, penalizzano la fruizione dei contenuti poiché è più facile che un firewall gestisca solo i pacchetti HTTP piuttosto che due tipi diversi di pacchetti (rendere un firewall capace di questo è una questione alquanto complicata).

Si è scommesso molto sulla terza modalità prima ancora che si parlasse veramente di overprovisioning, tuttavia in seguito utilizzare HTTP era sempre preferibile. Youtube utilizza la seconda modalità per via della facilità di gestione dei firewall e perchè così non è necessario che il riproduttore multimediale sia esterno al browser (come lo era Media Player) così da utilizzare uno stesso browser ed uno stesso riproduttore su qualsiasi architettura.
Il grafico di cui sopra costituisce, per così dire, una presa in giro: nella realtà il procedimento non può funzionare così per una questione di ritardo (che non può essere costante ma che anzi varia a seconda del traffico in atto e può essere imprevedibile) e di consegna dei pacchetti che, percorrendo strade diverse a seconda dei casi, possono giungere a destinazione in maniera non consecutiva e quindi disordinata. S'intende per **buffer di memoria** la distanza verticale tra i gradini delle ultime due "scalette" del grafico; il problema del buffer è che non sta scritto da nessuna parte quanto ce ne sia effettivamente, se sia capace di memorizzare tutti i dati trasmessi a parità di ritardo e di qualità/dimensione dei dati in base alle risorse disponibili.

Nella realtà si costruisce il contenuto multimediale, lo si spedisce in rete e quello che arriva allo storage è una schifezza praticamente ingestibile (dopotutto è così che funziona la rete): i dati così conciati sono inservibili e non possono passare al riproduttore multimediale a meno di essere ri-linearizzati come lo erano in fase di caricamento.

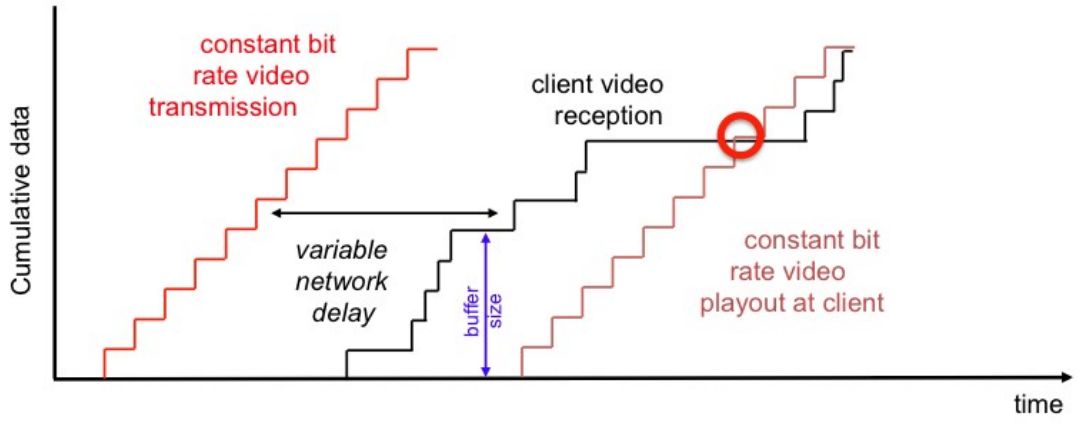


Stabilisco allora una pianificazione di come i frame vorrei venissero passati al riproduttore aggiungendo a ciascun frame del ritardo aggiuntivo; il **prefetch delay** costituisce quel ritardo aggiunto in modo tale da dare il tempo al buffer di riempirsi e di collezionare un po' di dati prima di consumare il contenuto multimediale nella maniera più fluida possibile. Il tutto è fattibile sfruttando due thread che collaborano tra loro così che il riproduttore non abbia bisogno di essere temporizzato.
Il **buffer** esiste perchè non si ha altra scelta, gestire il buffer costituisce l'unico modo che un'applicazione ha di rendere fruibile del contenuto soggetto a congestione. Il buffer immagazzina dati (i quali entrano come la rete concede) in una zona di memoria la quale prende in entrata il contenuto multimediale, lo comprime a dovere a seconda delle esigenze e lo restituisce secondo il profilo richiesto per essere decompresso e consumato. A lato client il buffer compensa le variazioni del tempo di trasferimento (il jitter); per esempio, Youtube possiede due soglie oltre le quali il video in esecuzione si blocca in attesa che il buffer venga nuovamente riempito (per evitare di svuotarlo completamente).

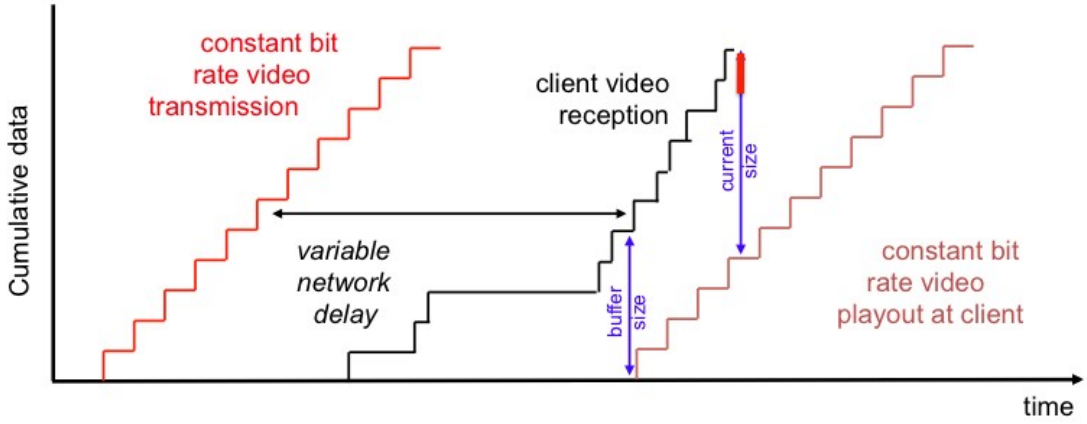


La dimensione del buffer però costituisce un nodo dolente: se è troppo piccolo il gioco tra la ricezione e lo smaltimento del contenuto diminuisce troppo e, se lo spazio è troppo poco, si rischia la perdita di dati che arrivano e non trovano spazio per immagazzinarsi; inoltre si corre il rischio di finire in *buffer overrun* e *underrun*. Se il buffer è troppo grande invece potrebbe rischiare di consumare troppe risorse del sistema operativo (e su client che non sono computer, come cellulari o televisori, l'economia di scala è molto importante poiché è sulla eventuale assenza di qualche capacità che il produttore abbatte il prezzo di produzione dell'apparecchio); il client potrebbe non averle nemmeno le risorse richieste e potrebbero introdursi dei ritardi veramente inaccettabili qualora il prefetch fosse troppo lungo.
Quando il buffer va in crisi la ricezione e la consumazione dei dati potrebbero diventare troppo o troppo poco vicine:

1. **buffer underrun**: si ha con tempi di ritardo estremamente lunghi durante i quali non si ricevono dati. Il guaio è che questi tempi d'attesa potrebbero finire col collidere con il grafico della consumazione del contenuto indicando con ciò lo svuotamento totale del buffer;



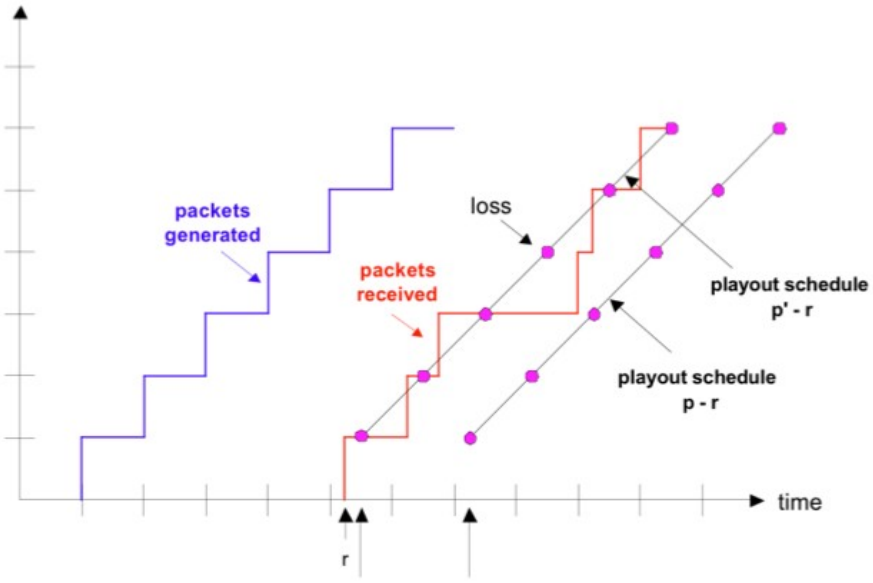
2. **buffer overrun**: si ha una raffica di arrivi di pacchetti in seguito ad un periodo di silenzio così da ritrovarsi a gestire in pochissimo tempo una curva di ricezione che improvvisamente impenna. Anche se di per sé sembra non creare troppi fastidi, se la cosa si verifica potrebbe darsi che la quantità del buffer richiesta ecceda quella effettiva causando di conseguenza lo scartamento di tutti quei pacchetti che arrivano ma che non trovano posto nel buffer (con l'aggiunta della beffa al danno, perchè quei pacchetti scartati hanno comunque consumato risorse di rete e di calcolo per essere, fondamentalmente, gettati via una volta arrivati).



Il buffer non è quasi mai scelto dall'utente ma riservarlo è compito del sistema operativo il quale fa uso di una classe dedicata allo scopo. Voler sfruttare meno spazio di quello riservato dal buffer è possibile ma non lo è in alcun modo ottenerne più di quanto deciso dal sistema operativo. Al massimo si può agire su cosa il server invia al buffer per poter arginare eventuali catastrofi: il buffer potrebbe mandare un feedback qualora lo spazio residuo scenda sotto un certo limite, allora si potrebbe chiedere al server di rallentare l'invio e di comprimere di più i dati multimediali (così da non scartare nessun pacchetto); purtroppo non è un procedura banale ed è il server a decidere quali sono le soglie.
Se i dati multimediali sono in realtime la questione si complica ancora di più: con questo tipo di dato non c'è alcun controllo sulla durata dello streaming, né sulla sua dimensione tantomeno sulla portata ed il profilo del traffico (non si può chiedere un bit rate più basso, perchè se la trasmissione è in realtime l'unico bit rate possibile è quello in atto!). Il profilo del traffico sarebbe anche predicibile se sapessimo a priori come funzionano player e codec, altrimenti si può provare a fare una predizione sul breve tempo.

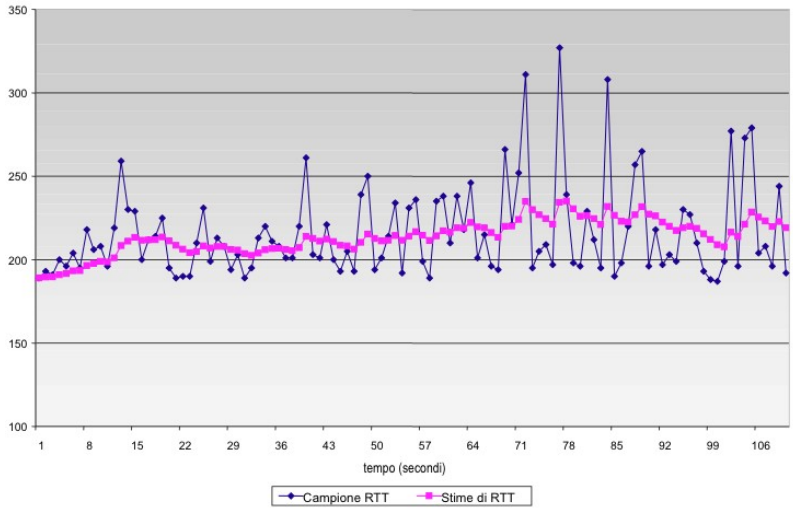
Caso di studio. Si può prendere come caso di studio quello della telefonia mobile, che è la più sensibile al traffico trasversale, sottoposta a vincoli molto stringenti per la fruizione dei contenuti (ma non del codec). La sorgente alterna periodi di voce e di silenzio del tipo [parola, silenzio, parola] e non si traccia mai il profilo del traffico della sorgente la quale genera traffico solo ed esclusivamente se necessario (per risparmiare risorse verranno mandati pacchetti solo se serve e non viaggeranno mai pacchetti vuoti, così per fare). Quando serve viene allocata banda per 64 kbps generando così raggruppamenti di pacchetti nemmeno tanto uniformi con un payload di 160 byte, uno ogni 20 ms (di conseguenza la banda per la voce è di 8 kBps). Ai pacchetti viene accorpato un header di livello applicativo e tutto viene poi incapsulato in un pacchetto UDP (non TCP per una questione di ritardo che nella telefonia sarebbe inaccettabile) il cui protocollo però ammette la perdita di qualche pacchetto, di tanto in tanto, a favore di una maggiore flessibilità e velocità. Si parla di **network loss** quando il pacchetto viene scartato a causa del buffer pieno; s'intende per **delay loss** lo scarto di un pacchetto che arriva troppo tardi e, poiché inservibile, viene gettato via (avendo tuttavia utilizzato ugualmente delle risorse); il **transmission loss** invece è la perdita di pacchetti dovuta al segnale elettromagnetico che non arriva a destinazione (oggi ha senso parlare di questo tipo di comportamento solo all'interno delle reti wireless). Il massimo ritardo tollerabile è (umanamente) di 400 ms oltre il quale fallisce qualsiasi call admission; se si sta sotto i 100 ms si riesce a parlare ed a fruire della conversazione in maniera abbastanza decente.
Oggi tutte le comunicazioni telefoniche sono VoIP, perchè nonostante la telefonata parta da un dispositivo fisso a livello di centralino il segnale passerà direttamente nella rete come se fosse sempre stato un pacchetto VoIP. Ma se è tutto VoIP perchè non si possono fare telefonate del genere direttamente da cellulare? Ovviamente per una questione economica, i gestori telefonici ci rimetterebbero dei soldi. Questo vale per tutta la telefonia, con tolleranza massima di ritardo che dipende dalla codifica dei dati (fino al 10% di perdita di dati è ancora accettabile per capire comunque quasi tutte le parole di una conversazione telefonica).
Il packet loss è l'impatto della percentuale di pacchetti persi che dipende da una serie di fattori e non è mai fisso, tra cui il tipo di codifica (alcune sacrificano la

qualità della voce per un'agevolazione in termini di velocità ed affidabilità, alcune codifiche sono specializzate per la gestione della voce parlata, altre per l'audio), la natura del contenuto multimediale ed il fatto che comunque sapremmo ricostruire il pattern a priori (anche se sappiamo cosa aspettarci dal player multimediale) così da alzare la tolleranza nelle perdite d'informazione.
Il **playout schedule** costituisce una pianificazione di tutte le tempistiche di passaggio dei pacchetti al riproduttore multimediale; il ricevente cerca di fare il playout di ogni pacchetto dopo Q millisecondi che questo è stato generato. Il ritardo della rete sommato a quello del buffer dev'essere uguale a Q; si tratta di un dato indipendente dal mezzo fisico di trasmissione. Il pacchetto con valore di timestamp pari a T dovrebbe essere passato al decoder al tempo T+Q e se il pacchetto arriva oltre il tempo T+Q allora i dati è come se siano stati persi. Bisogna trovare un buon compromesso per stimare Q, che sicuramente deve essere grande abbastanza per evitare il delay loss frequente (quindi la perdita di numerosi pacchetti) ma non troppo per evitare di sacrificare troppa interattività (laddove la trasmissione risulti interattiva). L'ideale sarebbe che Q si adattasse da solo ad ogni circostanza; per fissare ed implementare Q dopotutto basterebbe un buffer “intelligente” di lunghezza opportuna (che riordini tutti i pacchetti a seconda del T) ed un sistema per estrarli “al momento giusto”.



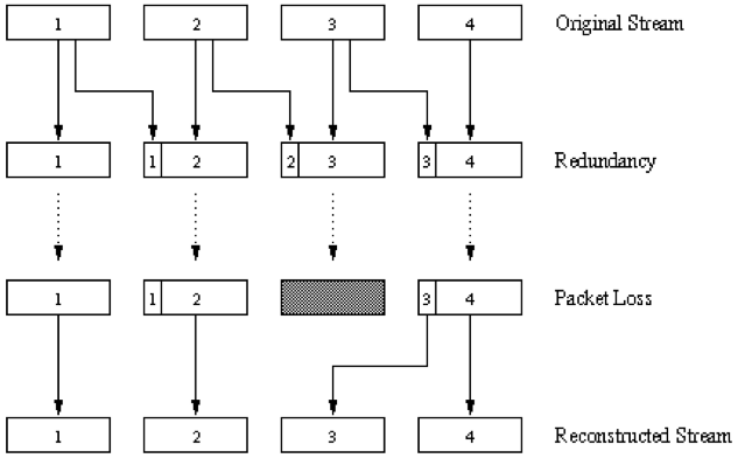
Nello schema sopra la prima retta obliqua che s'incontra a partire da sinistra identifica il comportamento del playout nel caso di un buffer di dimensione pari a zero (quindi nullo): ad un certo punto s'incappa in un delay loss. Il profilo di ricezione (la linea spezzata rossa) deve sempre trovarsi per la maggior parte dalla parte sinistra rispetto alla retta del playout perchè il sistema funzioni e non ci siano perdite di alcun genere.
Il **playout adattivo** indica l'adattamento di Q in maniera interattiva, dinamica ed autonoma. Si tenta di minimizzare sia Q che il numero di pacchetti che arrivano dopo l'istante di schedulazione. Si deve avere la massima sfiducia nel buon comportamento della rete, così dopo la ricezione dei primi pacchetti del contenuto multimediale si prova a calcolare il ritardo nella rete, quindi si calcola Q in maniera che tenga conto del ritardo end-to-end (ritardo tra generazione e ricezione del contenuto) tanto da aggiustarla di volta in volta ad ogni misurazione del ritardo di rete. In telefonia per ovviare al problema si inviano pacchetti eliminando la pausa di silenzio tra le parole (va tutto bene, basta che i

campioni vengano inviati in maniera costante, nei momenti di massima attività si cerca comunque di mandare in playout i pacchetti con tasso costante ovvero ogni 20 ms). Ci si dimentica però di dire che anche se dati come audio e video viaggiano su flussi di natura diversa (audio e video non sono equamente importanti per una buona fruizione del contenuto, l'audio lo è più del video) bisogna tenere conto della sincronizzazione; non si può fare con la musica e la stima del ritardo end-to-end è difficilissima ed estremamente variabile (oltre ad essere praticamente impossibile salvo sincronizzazione perfetta degli orologi di sistema dei due capi della rete). Tale valore risulta essere uno dei più instabili al mondo: per stimarlo a livello di applicazione si fa come lo fa TCP, ovvero sfruttando la tecnica del filtro semidigitale od a coda. Tale tecnica è ottenuta calcolando il ritardo medio fino a che non si verifica un ritardo ancora maggiore (o minore); si fa allora una media tra il nuovo ritardo e la media dei vecchi ritardi tenendo in maggior considerazione i ritardi più recenti e meno quelli più vecchi nel tempo. In pratica si cerca una media basata su eventi passati con l'attenzione ai campioni più recenti, la quale verrà modificata di volta in volta nel momento in cui si riceve un nuovo valore. Un *fattore alfa* indica quanto pesa una vecchia misurazione rispetto ad una nuova. La stessa cosa avviene con il Round Trip Time (RTT) perchè varia a seconda della congestione della rete ed anche in questo caso è impossibile da misurare accontentandoci di fare una stima con una media esponenziale; in certi casi capiterà che tale stima venga ripetuta più e più volte in una data unità di tempo, il guaio sta nel fatto che la stima stessa comprende l'esecuzione di due operazioni di moltiplicazione computazionalmente onerose in termini di tempo e risorse, cosa che si può migliorare sostituendo una moltiplicazione con una sottrazione così da alleggerire il carico della CPU: per far ciò il fattore alfa dev'essere il più vicino possibile al valore di 0,125.
La stima è completamente disgiunta dai campioni (grazie ai quali è possibile la stima) tuttavia serve per catturare e fissare il trend dell'andamento delle prestazioni dove il primo campione coincide con il primo valore di media.



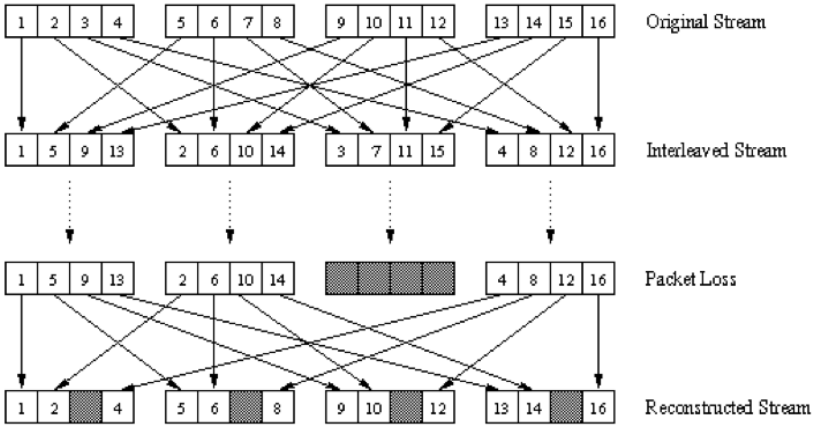
Dimensionare Q in questa maniera ha un significato solamente spannometrico perchè non si assicura alcuna protezione duratura da eventuali delay loss (cosa che avviene solo ed esclusivamente se tale valore verrà moltiplicato per un fattore di un certo tipo, almeno del 50%).
Anziché ritrasmettere i pacchetti andati perduti come avviene spesso in UDP (la

ritrasmissione non è un opzione percorribile, altrimenti potremmo usare direttamente TCP anziché UDP) si preferisce recuperare le informazioni perdute a livello di codifica, interpolando il pacchetto precedente e quello successivo a quello effettivamente smarrito in modo da creare un'informazione mediana che possa ingannare il fruitore del servizio, tuttavia non sempre è possibile. Bisogna utilizzare dei meccanismi di **FEC** per introdurre un po' di ridondanza dei dati: si tratta di informazioni che vengono aggiunte solo dopo l'estrapolazione dal supporto del contenuto e prima della sua effettiva trasmissione e fruizione al lettore (in questa maniera non si vanifica il lavoro di compressione e codifica di MPEG), il volume del file aumenta così del 5%.
Nella **FEC semplice** si manda un pacchetto extra ogni N pacchetti inviati (per esempio, ogni 10 pacchetti) occupando di conseguenza la banda con un EXOR dei bit di tutti gli N pacchetti già inviati: in questa maniera qualsiasi pacchetto perso tra i 10 iniziali dell'esempio può essere ricostruito tramite l'interazione dell'undicesimo pacchetto con i nove rimanenti. Se ho davvero N pacchetti già in ballo spreco una quantità di banda pari ad 1/N in più rispetto a quella totale riservata al servizio, inoltre se N è molto grande ci si può permettere di perdere al massimo un solo pacchetto (se ne sfuggissero due si parlerebbe di almeno 2 EXOR e conseguentemente di 2/N quantità di banda da sommare a quella già riservata, e via discorrendo); aumenta di conseguenza il tempo di playout e se davvero si perde un pacchetto bisogna attendere che arrivi il pacchetto EXOR e che si ricostruisca il pacchetto mancante aggiungendo nuovo ritardo alla trasmissione che deve comunque essere compreso del tempo Q che dev'essere almeno superiore al tempo necessario per ricevere N+1 pacchetti. Allora il valore di N va scelto con cura sapendo che all'aumentare di N diminuisce la banda sprecata, aumenta il ritardo di riproduzione ed aumenta la probabilità di perdere 2 o più pacchetti in una sola sequenza. Sostanzialmente si tratta di un meccanismo veramente poco utilizzato, praticamente per nulla.



Molto più utilizzato è il metodo **piggyback**: si aggiunge ridondanza al pacchetto successivo (quasi come se si aggiungesse dell'informazione “sulla sua schiena”) in modo da recuperare qualsiasi eventuale perdita; tipicamente l'informazione aggiunta costituisce la stessa del pacchetto precedente ma molto più compressa. Si manda il primo pacchetto, poi il secondo al quale viene allegata l'informazione compressa del primo, ed avanti così: funziona solo se non ci sono due perdite consecutive (a meno che non si usi uno schema che considera gli ultimi N pacchetti). Non è un metodo perfetto, ma è fruibile tant'è

che viene usato molto in telefonia. Se avviene la perdita di due pacchetti consecutivi si avrà comunque un "buco" nella trasmissione.



L'**interleaving** costituisce una soluzione non sempre utile (meglio utilizzarla con uno stream audio): in questa maniera i dati vengono suddivisi in unità più piccole dette *chunk* (ogni dato viene spezzettato in sotto-unità tutte simili, per cui il dato 1 sarà suddiviso in unità 1.1, 1.2, 1.3 e così il dato 2 in 2.1, 2.2, 2.3 ecc.) ed ogni pacchetto è costituito da una serie di unità che proviene da campioni diversi (quindi il pacchetto 1 sarà costituito da tutte le prime parti di ogni dato spezzettato del tipo 1.1, 2.1, 3.1 ed il pacchetto 2 conterrà 1.2, 2.2, 3.2 ecc.), in questa maniera anche se un pacchetto va perso si ha la quasi totalità del resto delle informazioni. L'orecchio tollererà meglio la perdita di una infinitesima parte dell'informazione piuttosto della perdita di un intero blocco: la perdita di informazione verrà allora mimetizzata poiché distribuita su un periodo più lungo. In questa maniera non si introduce alcun tipo di ridondanza e ci si può permettere di perdere diversi pacchetti più o meno in successione, tuttavia aumenta il ritardo di playout perchè per ricostruire lo streaming devo attendere di ricevere tutti i pacchetti. In questa maniera è facilitata (all'occorrenza) un minimo di crittografia poiché il rimescolamento dei chunk potrebbe non essere casuale ma ottenuto dall'utilizzo una qualche chiave pubblica. Il segnale satellitare cifrato funziona proprio così: si spezzano le linee di scansione in punti random decisi da una funzione presente sulla smart card inserita nel decoder con conseguente inversione dei due segmenti spezzati.

Protocolli di segnalazione

Vengono sfruttati dalle applicazioni per gestire un flusso multimediale. Per fare il setup della comunicazione si utilizzano i **protocolli di segnalazione** (dei quali ne guarderemo solo alcuni, quelli più orientati alla comunicazione, poiché ne esistono vari) così da coordinare meglio i flussi di dati. I protocolli che gestiscono il flusso durante il suo funzionamento sono RTP/RTCP mentre quello che lo prepara prima della partenza è SIP.

RTP/RTCP. Sono due protocolli per il controllo del flusso multimediale i quali permettono, a livello applicativo, l'implementazione di un valido sistema di reportistica.

1. **Real-Time Protocol** → produce pacchetti TCP con header TCP per il trasporto di statistiche e di dati audio e video incapsulati all'interno di un

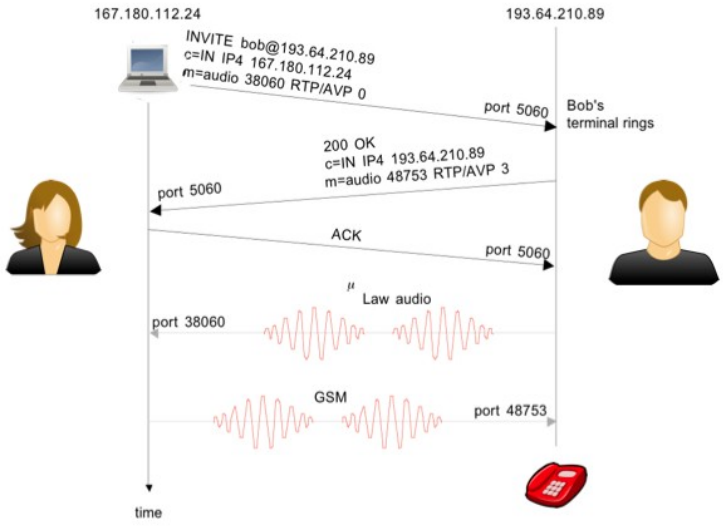
pacchetto UDP.

2. **Real-Time Control Protocol** → stabilisce nella pratica un formato per i report scambiati tra sorgente e destinazione o viceversa.

Le statistiche sul servizio si calcolano solitamente all'arrivo dei pacchetti ma non sono significative per chi riceve quanto per chi manda lo stream di dati. Con RTCP periodicamente si aggiornano le statistiche presso la sorgente. I report vanno comunque in entrambe le direzioni: in realtà RTP è più usato a livello di applicazione perchè aggiunge un certo ritardo nello stack protocollare, cosa di cui è necessario tenere conto nel calcolo finale del ritardo della comunicazione (spesso ignorare questo fatto non fa tornare i conti): si pensi che il ritardo aggiunto da uno stack all'interno di un sistema operativo è di 20 ms ad ogni giro per un totale di 40 ms complessivi (andata + ritorno) → in Linux il sistema di schedulazione fa in modo che i pacchetti arrivino con un jitter pari a zero. RTP implementa un processo a parte che ascolta la rete e compie delle statistiche in maniera indipendente; si occupa anche di raccogliere statistiche distribuite tra più destinazioni rapportate ad un unica sorgente, tuttavia RTP ha enormi problemi implementativi: non è puramente realtime, tuttavia supporta le trasmissioni realtime.
RTP non fornisce nessun tipo di supporto alla qualità di servizio: aggiunge solo un identificatore di payload, un numero di sequenza, un timestamp ed un identificatore di sorgente (per compiere agevolmente statistiche su sorgenti differenti). Viene gestito unicamente dai nodi finali (end-system) ed i router lungo il percorso lo ignorano completamente vedendolo transitare come un banale pacchetto UDP.
Anche RTCP non garantisce nessuna forma di qualità di servizio, tuttavia i dati che trasporta possono essere utili alle applicazioni per gestire una sorta di QoS fornendola, indirettamente, con altri mezzi: numero di pacchetti inviati, numero di pacchetti persi, tempo medio di interarrivo e jitter stimato (anche se il jitter è per definizione un dato alquanto labile). Periodicamente sorgente e destinazione si scambiano dei messaggio RTCP contenenti gli aggiornamenti dei parametri trasmissivi. RTCP si sfrutta per sincronizzare i flussi all'interno di una stessa sessione RTP (un messaggio RTCP contiene informazioni sull'allineamento dei timestamp di tutti i flussi relativi alla sessione); avendo a che fare con contenuti realtime può esserci una certa asincronia alla sorgente così l'informazione relativa al timestamp può essere utilizzata a lato ricevente per coordinare l'estrazione dei dati dai singoli buffer. RTCP non può avvalersi più del 5% della banda totale destinata/riservata alla trasmissione RTP (ed il controllo è tutto demandato ai nodi di destinazione che si servono di appositi contatori).

SIP. Acronimo ed abbreviazione di **Session Initiation Protocol**, costituisce uno standard IEEE. Costituisce un protocollo di consegna finalizzato alla comunicazione tra due entità che si scambiano informazioni multimediali e permette sostanzialmente di effettuare una chiamata, identificando il chiamante ed il chiamato, per poi abbandonare la conversazione. Gode di una visione su lungo periodo: poiché le telefonate si faranno (e già si fanno) via Internet, gli utenti saranno identificati tramite nome ed email (e non più via numero di telefono): solo in questa maniera si potrà raggiungere chiunque in qualsiasi momento ed in qualsiasi luogo esso si trovi. Nella telefonia mobile ad identificare un utente è il numero identificativo della carta SIM che serve ad autenticarci nella rete; esistono quindi server che associano gli ID delle SIM con i relativi numeri di telefono in modo poi da rendere possibile l'istituzione di un

canale comunicativo tra due SIM differenti. Il numero di telefono (che una volta era prettamente un insieme di segnali passati ai centralini perchè venisse istituito in canale comunicativo tramite circuito fisso) continua ad essere utilizzato poiché in certe parti del mondo la call setup è fatta ancora in maniera tradizionale.
SIP non rappresenta in alcun modo un protocollo lungo tutto lo stack ma solo a livello applicativo. Grazie a SIP si potrebbe raggiungere chiunque ovunque ma oggi ogni apparecchio è contraddistinto da un identificativo differente, rendendo l'attuazione del sistema alquanto difficile (ma non impossibile).
SIP può essere sfruttato ANCHE per la telefonia, ma non solo (anche il testo può essere gestito da SIP, come accade in Google Talk, ed il video).
Durante una call setup si contatta il chiamato, si contratta una codifica comune per le informazioni da scambiarsi con l'ausilio di un intermediario super partes che metta d'accordo i due nodi ed infine si chiude la telefonata (anche e soprattutto per una questione di gestione delle risorse e di tariffazione). Il setup si occupa di mettere in piedi il canale di comunicazione, in seguito la chiamata sarà gestita da un protocollo di controllo dedicato allo scopo (il quale si dedica alla competizione per le risorse aggiungendo e togliendo flussi multimediali alla chiamata, al cambiamento della codifica condivisa, all'eventuale aggiunta di nuovi partecipanti alla conversazione, alla messa in attesa ed al trasferimento della chiamata).



Call setup. Alice conosce già l'indirizzo IP di Bob, allora apre una socket e manda un messaggio al suo interlocutore allegando alcune informazioni sulla creazione della chiamata ed il suo utente: tali informazioni includono il tipo di richiesta mandata (in questo caso INVITE), la persona a cui sono mandate, che sarebbe Bob, ed il suo indirizzo IP, quindi una serie di comunicazioni riguardo la versione IP utilizzata (IP4), l'indirizzo IP di provenienza, il tipo di dato comunicato (che è audio), il numero della porta presso la quale avviene la comunicazione ed il protocollo, o la famiglia di protocolli, che Alice è disposta ad utilizzare nella comunicazione con Bob; la sintassi del messaggio è molto simile a quella di HTTP perchè facilmente adattabile ai file. L'informazione allora arriva a Bob e lui risponde con un secondo messaggio di tipo OK (come del resto farebbe un server web HTTP, per dire che la comunicazione allora può essere aperta sottostando alle richieste di Alice), in aggiunta ad informazioni

come la conferma del suo indirizzo IP, il tipo di dati che vuole scambiare ed i protocolli di cui dispone a sua volta (specificando comunque una preferenza). La contrattazione del codec va effettuata perchè le architetture telefoniche sono tutte proprietarie e pagano delle royalty, si tratta di una questione molto spinosa: Alice è tenuta a dire a Bob di quali protocolli dispone e Bob dovrà risponderle con l'insieme delle codifiche supportate a sua volta perchè potrebbe preferirne altre o semplicemente potrebbe averne di diverse a disposizione (altrimenti Bob accetta le codifiche di Alice e non ci sono ulteriori problemi). I messaggi SIP possono essere mandati sia su UDP che su TCP (RTP) e la porta di default è la 5060. Se i due insiemi di codifiche risultano essere disgiunti la call admission fallisce, altrimenti si usa uno dei punti (o protocolli) in comune. Le royalty sono pagate per l'encoding ma non per il decoding, così spesso l'insieme di protocolli di codifica sono ristretti mentre quelli di decodifica sono assai più ampi: l'importante è che Bob decodifichi uno dei protocolli di Alice e che Alice faccia altrettanto con Bob, allora la telefonata funzionerà comunque benissimo.

Call negotiation. La negoziazione è fatta tramite UDP anche se si preferisce TCP perchè tende a confondersi più facilmente con i pacchetti HTTP così da non destare sospetti nei firewall. Se Bob non accetta nessuna delle codifiche di Alice risponderà con una messaggio negativo (606 Not Acceptable) ed una sua lista dei codec supportati. Alice allora potrà mandare un altro messaggio d'invito specificando una delle codifiche supportare da Bob.

Call rejecting. Bob, oppure un suo intermediario per conto suo, rifiuta la chiamata allegando ovviamente un motivo di tale rifiuto: canale occupato, pagamento del servizio richiesto, impegnato in un'altra comunicazione oppure non raggiungibile.

Il chiamato è identificato dal nome e dalla propria email. Con il solo indirizzo email non si riesce ad avviare la chiamata salvo se è già stato associato a priori ad un indirizzo IP stabile. I primi portatili si avvalevano di un **mobile IP**: c'era un agente domestico (home agent) che faceva da intermediario sull'indirizzo IP della macchina. Su una rete diversa da quella domestica appena ci si connetteva con l'apparecchio associato ad un certo indirizzo IP l'agente locale si metteva d'accordo con l'agente domestico in modo da inoltrare a quel dispositivo qualsiasi chiamata pervenuta a quello specifico indirizzo IP anche se non si trovava nella rete domestica (*routing triangolare*). In questa maniera si avevano diversi guai con i firewall tuttavia senza un intermediario è impossibile creare qualche cosa di realmente *mobile*. Gli intermediari di SIP sono:

1. **SIP Registrar.** Server di competenza dell'operatore telefonico (che quindi si occupa di fatturare le chiamate) presso il quale è possibile registrarsi per rendersi disponibili e reperibili. All'accensione (ed in maniera periodica durante la telefonata) il telefono manda un messaggio al SIP Registrar notificando il suo indirizzo IP attuale;
2. **SIP Proxy.** Si tratta del server attraverso il quale avvengono tutte le chiamate: il messaggio d'invito viene inoltrato al proxy il quale si occupa di farlo recapitare al destinatario (a volte tramite una catena di proxy), è lui che conosce l'indirizzo del destinatario o che comunque lo cerca, l'indirizzo non va per forza comunicato a priori. Una volta trovato il destinatario il proxy passa la palla al suo registrar. Il destinatario allora risponde attraverso la stessa catena di proxy ma includendo alla risposta

il suo personale indirizzo IP, a seguito di ciò i due terminali potrebbero comunicare direttamente (cosa che non avviene salvo nel caso di una rete locale, altrimenti non sarebbe possibile fare un'adeguata fatturazione del servizio).

Solitamente proxy e registrar sono unificati ed uniti in un solo centralino. Nessuno ci obbliga ad avere lo stesso indirizzo IP sulla stessa LAN e quindi un solo canale d'uscita sul register, soprattutto a farlo solamente in WiFi. L'operatore telefonico preferisce far pagare di più per i servizi di voce piuttosto che quelli di traffico di dati. Proxy e registrar sono intermediari positivi. SIP è più conveniente di Skype che non è puramente VoIP e si possono creare dei filtri per scremare e selezionare le telefonate. Il VoIP si appoggia su UDP e possono venire a crearsi delle difficoltà tecniche quando si accede ad una rete *nattata* (sotto protocollo NAT) perchè non si è capaci di essere rintracciabili all'interno della rete nattata salvo facendo uso di qualche strana porcheria; si costruisce allora un sistema che, avvalendosi di socket che rimangono in piedi per tutta la durata della chiamata, mandano il segnale di chiamata anche nel senso opposto scavalcando agevolmente il NAT.

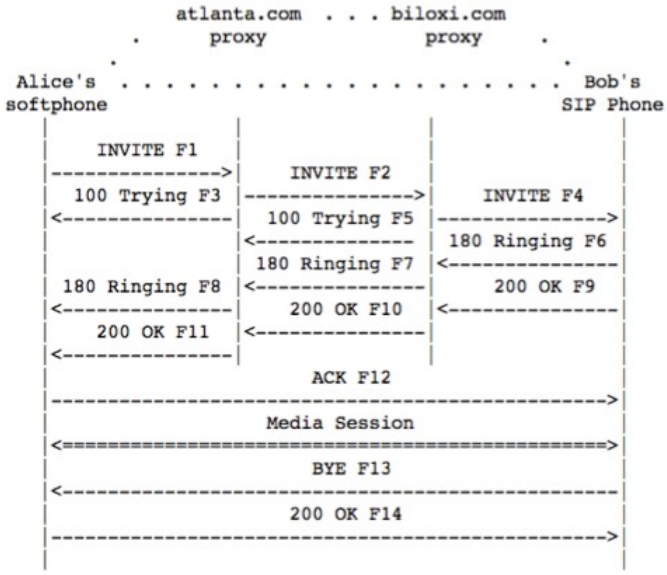


Figure 1: SIP session setup example with SIP trapezoid

Alice (tramite un'applicazione su computer che simula un telefono SIP) manda una richiesta d'invito a Bob (il quale si serve di un reale telefono SIP fisico): tra di loro ci sono ben due proxy che fanno da intermediari. Bob restituisce un messaggio di “sta squillando” ma che potrebbe essere anche di “linea occupata” (allora in questo caso la connessione cade per liberare immediatamente le risorse senza sprecare altro tempo). Se invece tutto va bene e la via è libera Bob alza la cornetta mandando un messaggio di OK ad Alice.

STUN Server. Acronimo di *Simple Traversal of UDP through NATs*. È un server che permette di gestire un client all'interno di una rete che usa NAT: tipicamente un server di questo tipo possiede almeno una interfaccia di rete con indirizzo pubblico. Costituisce una sorta di buffer a metà strada che tenta di attutire gli effetti del jitter (ma non serve strettamente alla telefonata). Un terminale prende accordi con lo STUN server per poter ricevere le chiamate: tipicamente lo STUN converte i messaggi di invito in UDP e li rigira su una

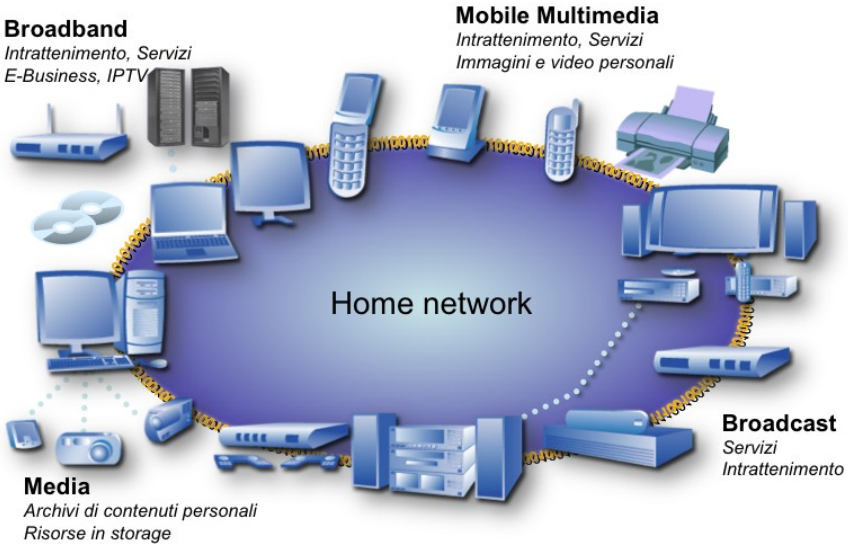
socket TCP che in terminale mantiene aperta con il server.

Asterisk. Quanto detto sin'ora si può tranquillamente attuare via software: *Asterisk* implementa un centralino telefonico PBX ed in più gestisce l'hardware sottostante dando prova di grandi funzionalità ad un costo abbastanza basso poiché si tratta di software libero e di larga adozione per le piccole realtà. L'unico problema è la sua mancanza di un interfaccia grafica che rende difficile l'interazione con almeno 60 file di configurazione (per esempio, bisogna segnalare tutti gli identificativi dei dispositivi ammessi in rete come numeri di telefono, indirizzi IP ed altro, quindi segnalare le direttrici d'uscita del traffico, a lato client vanno segnalati i proxy ed i registrar a cui interfacciarsi, ecc). Asterisk definisce un diningplan (??), ovvero quando un server manda una sequenza di numeri associa il telefono al segnale e viceversa.

Un'alternativa a SIP è **H.323** che costituisce un standard tuttavia poco accessibile. In questo caso si parla di una suite completa che copre tutto, compresi admission control e codec per la telefonia (e non per il testo): molto più complesso di SIP da gestire ed implementare.

Digital Living Network Alliance (DLNA)

Si tratta di un prodotto vero e proprio proveniente da aziende che fabbricano elettrodomestici e non di un caso di studio (la consultazione delle specifiche infatti è a pagamento). Costituisce un ecosistema per il trasferimento di dati multimediali tra dispositivi diversi ma in maniera organizzata (il computer, in questo caso, va considerato più come un insieme di hard disk che possono affacciarsi sulla rete grazie ad una scheda di rete e meno come un calcolatore). Moltissime aziende ne fanno parte così da far comunicare tra loro i propri prodotti qualora aderiscano a certe determinate specifiche. La Apple è una delle poche aziende tecnologiche a non aderire al consorzio preferendo mandare avanti un progetto personale. Fondamentalmente si tratta di un consorzio all'interno del quale si uniforma e si rendono integrabili software ed hardware diversi per natura e per destinazione.
Lo streaming a volte può essere a “corta distanza” allora si vorrebbe poter condividere i contenuti multimediali tra i dispositivi che sono già presenti nell'ambiente domestico. Però, si sa, gli utenti pretendono molto: facilità di installazione e di utilizzo (possibilmente con un solo grande pulsante); ampia compatibilità anche e soprattutto con i dispositivi già presenti ed acquistati precedentemente (problema prettamente economico); mentre non interessa più di tanto la fusione tra le funzionalità se il funzionamento complessivo risulta essere abbastanza fluido ed organico. A questo punto non si tratta più di fare solo dello streaming ma anche di interagire con qualsiasi contenuto; lo streaming passa in secondo piano, perchè il problema maggiore a questo punto è studiare un protocollo di controllo universale, noto a tutti i dispositivi, che tutti possano implementare e che possa tollerare l'interazione tra versioni dello stesso non sempre nuovissime ed aggiornate (certe volte pensare di fare un upgrade continuo è semplicemente fuori discussione, magari non è nemmeno preventivato, per motivi anche economici).
Il consorzio raggruppa 270 e più aziende che si occupano di connettività e di fornitura di servizi, produttrici di hardware e software per la produzione e la fruizione multimediale.



Il range di dispositivi con cui si può aver a che fare in questa circostanza è molto ampio e si rischia di fare confusione.

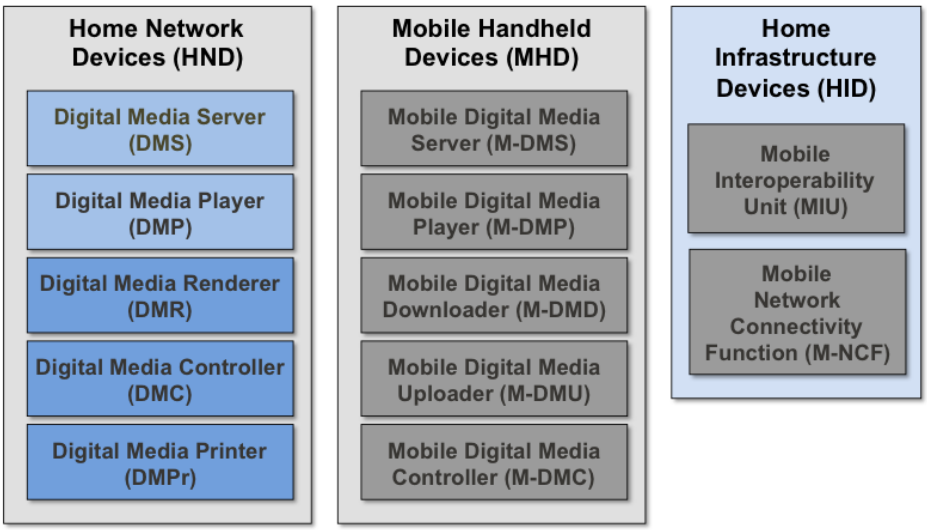
1. **Media.** Dispositivi dedicati all'intrattenimento, all'erogazione di servizi di E-Business e IPTV, spesso costituiscono e rappresentano delle memorie di archiviazione all'interno della rete domestica (il software deve pur recuperare i dati da qualche parte), altrimenti bisogna gestire le interazioni con l'esterno e le connessioni di rete.
2. **Broadband.** È utilizzato per accedere a diversi contenuti, magari pagando un abbonamento, in maniera distribuita (bisogna comunque adattare i servizi in abbonamento ad essere fruiti in questa maniera lavorando con i loro protocolli chiusi e molto proprietari), così però bisogna tradurre le codifiche dei dati provenienti dall'esterno verso l'interno (allora ci si potrebbe chiedere quanto dev'essere potente la CPU del router?).
3. **Mobile Multimedia.** La disponibilità dei contenuti però non va legata alla natura fissa o mobile di un dispositivo: wifi e mobilità non sono sinonimi. I dispositivi mobili inoltre vanno registrati ed autenticati ogni volta che lasciano e poi tornano nella rete domestica.
4. **Broadcast.** Rappresentano servizi fruibili in modalità da 1 verso tutti come tv e radio. Può capitare che qualche volta sia on demand, altre invece il flusso sarà tutto broadcast.

Il DLNA si prefigge la definizione di framework di interoperabilità comunque per i dispositivi su reti domestiche di piccole dimensioni all'interno delle quali tutti i dispositivi parlano con tutti rispettando certe regole per l'interattività anche nei confronti di dispositivi più contenuti con batterie e CPU più piccole e meno efficaci; la definizione di “linee guida” basate su standard aperti e, dove possibile, già esistenti per favorire l'adesione su larga scala dei produttori (anche di coloro che lavorano a progetti Linux); identificare dove c'è necessità di introdurre nuovi standard; interoperabilità a livello di formato; definizione di una qualità di servizio standard per tutti basata su una QoS condivisa (perchè se ne esistono di diverse è come se non ne esistesse nessuna); accelerazione e penetrazione del mercato tramite “compliance tests” e loghi di compatibilità sui dispositivi.
DLNA sino ad oggi ha visto diverse versioni, la corrente è la 1.5: quello che si

trova su mercato oggi è frutto di lavoro e ricerca del 2005/2006 tuttavia il ciclo per il lancio di un nuovo televisore è lungo dai 5 ai 7 anni, tra gli investimenti delle aziende, i test e l'avvio delle vendite. Si parla di allargare i formati per ottenere molti più contenuti già codificati rilassando dalla parte dei dispositivi il supporto di base di alcuni decodec.

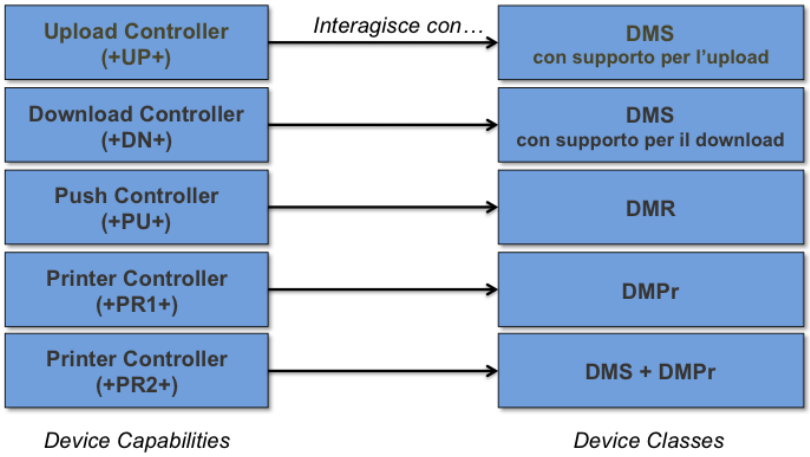
- Versione 1.0
  - Specifiche rilasciate nel 2005, certificazioni dal 2006
  - 2000+ dispositivi certificati (al 2008)
  - 2 volumi: "architetuttura e protocolli" + "media formats"
  - 2 classi di dispositivi e circa 50 formati
- Versione 1.5 (specifiche a pagamento)
  - Espansione dell'ottobre 2006, certificazioni dal 2007
  - 9000+ dispositivi certificati (al 01/2011)
  - 3 volumi: aggiunta "link protection"
  - 12 classi di dispositivi, 5 "device capabilities" e circa 250 formati
- Versione 2.0
  - In fase di rilascio (?)
  - Sincronizzazione di contenuti
  - Altri formati
  - Scheduled recording
  - Gestione del DRM

I dispositivi si dividono in **classi** (device class) che altro non sono se non unità certificate dove le certificazioni sono ovviamente rilasciate dal consorzio. Un dispositivo può avere una funzione certificata ma non è detto che lo siano anche le altre, altrimenti più certificazioni possono risiedere su uno stesso dispositivo in merito a più caratteristiche distinte (quindi un solo dispositivo può ricadere in più classi diverse). Una **categoria** (device category) è invece un insieme di classi che condividono caratteristiche simili (come la mobilità, oppure il funzionamento senza alimentatore, ecc); a certe classi si possono aggiungere poi altre caratteristiche (device capability) costituite da una **componente** che può essere aggiunta per aumentare certe funzionalità: esistono solo in relazione ad una classe e non possono esistere in maniera autonoma.



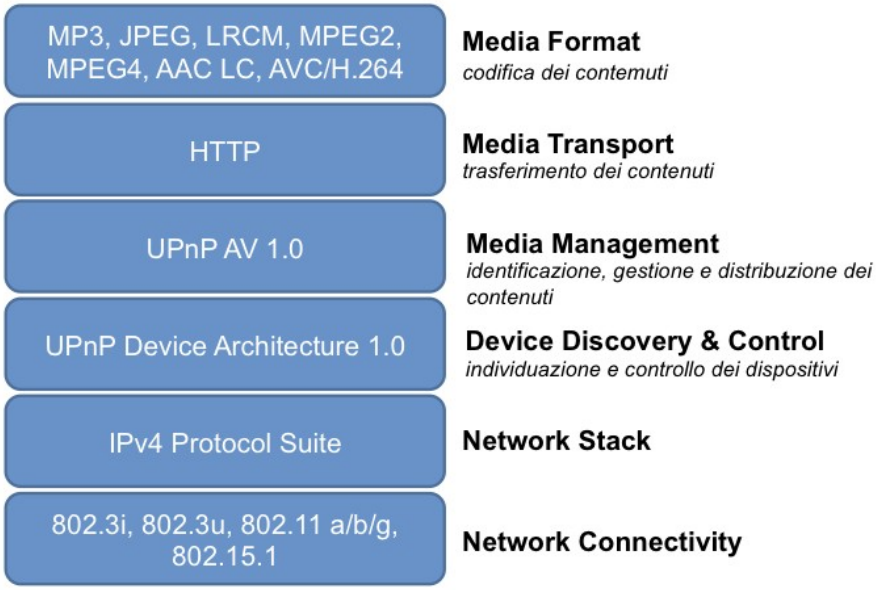
La categoria Home Network Device costituisce la categoria dei dispositivi

agganciabili ed è suddivisa in diverse classi: importa poco la differenza tra tipi di dati multimediali (come audio e video) così ci si concentra sulla differenza tra quale dispositivo li consuma (*player*) e quale li genera e li eroga (*server*). Il *player* ed il *renderer* sono oggetti diversi e disgiunti: la tv può essere intesa come renderer quando il player può identificarsi nel decoder satellitare. Possono interfacciarsi più player sullo stesso renderer. Il *controller* si occupa di tenere d'occhio l'esecuzione dei processi mente il *printer* può essere una stampante che rende "fisica" una fotografia od un documento di testo (può essere inteso anche come un renderer anomalo perchè crea una copia concreta di qualunque flusso di dati coerenti con la stampa). La differenza tra renderer e printer può risiedere nel fatto che un renderer è tale a prescindere da tutto mentre il printer dipende dalla presenza o meno di altri fattori come, per esempio, la carta.
La categoria Mobile Handheld Device costituisce il gruppo dei dispositivi mobili che vanno autenticati ogni volta che accedono alla rete domestica (dopo averla precedentemente lasciata). Il *downloader* e l'*uploader* comunicano con il sistema domestico per la gestione ed il traffico delle informazioni multimediali; il *controller* invece funge da "telecomando".
Per la coordinazione tra la rete domestica e l'esterno senza toccare nessun dispositivo in maniera diretta esiste la categoria degli Home Infrastructure Device. La MIU autentica i dispositivi per l'interazione nell'ecosistema mentre l'M-NCF si dedica alla connettività wireless trasparente.

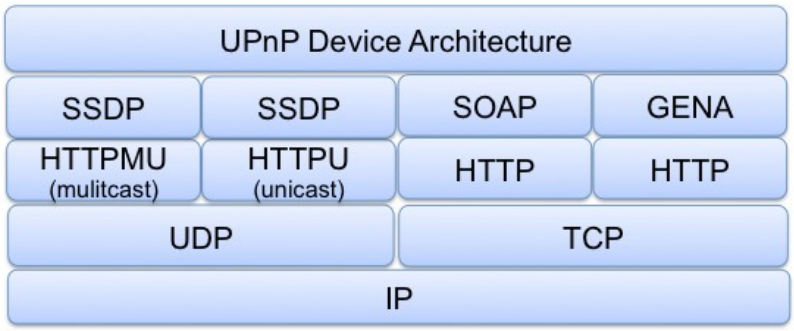


DLNA implementa anche uno schema per le interazioni, detto interoperability framework: le componenti interagiscono con le classi (anche se non è detto che il rapporto d'interazione debba essere per forza 1:1). Bisogna ricordare che tutte queste tecnologie esistono già, l'unica cosa che manca è un unico dispositivo con un solo grande bottone che faccia da solo tutto il lavoro.
Come i singoli elementi/dispositivi parlano tra loro a livello di infrastruttura lo definisce il formato per la codifica dei contenuti (media format → che fa parte della famiglia MPEG ed è parametrizzato, quindi si basa sull'utilizzo di frame rendendo meno difficile la perdita e la frammentazione di una sola parte o di tutta l'informazione); il livello di trasporto per il trasferimento dei contenuti si basa su HTTP (quindi TCP e non UDP) per via della banda sempre più larga e dei sempre più bassi tempi di risposta (cosa che avviene già su molti dei dispositivi che esistono già e che sono già stato venduti da anni); UpnP attua il riconoscimento dei dispositivi connessi e lavora interfacciandosi con il NAT

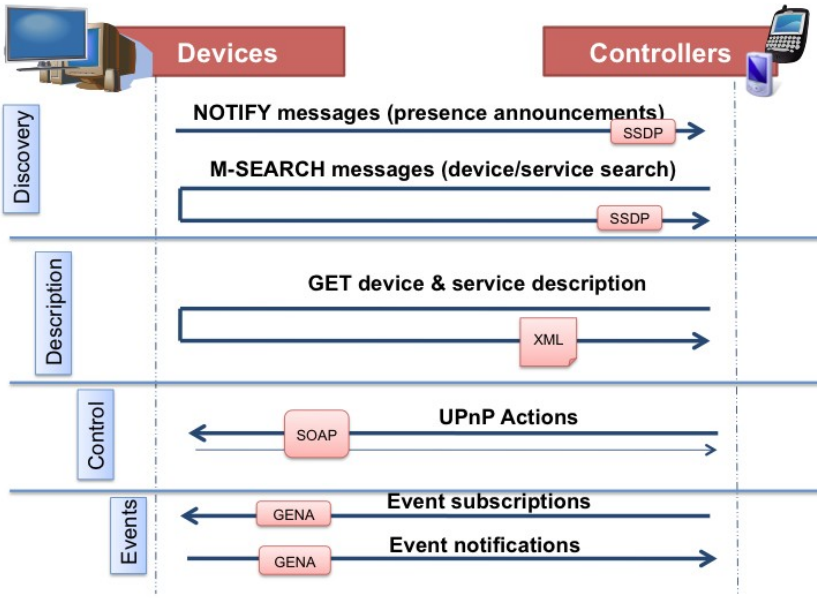
qualora sia richiesto a livello di media management così da istruire il router che ciò che arriva dalla rete va inviato direttamente ad un certo dispositivo, come la tv (UpnP possiede due anime diverse a seconda dell'uso che se ne deve fare, va installato sul router e sul dispositivo perchè funzioni al meglio); l'unica rete supportata è Ipv4 ma non è detto che non sia forzatamente unicast (può essere anche multicast, allora dev'essere supportata); ci sono poi vincoli su quale tecnologia usare a livello di data link (come il wireless o altro).



Universal Plug and Play (UPnP). Costituisce una suite di protocolli (quindi non è un protocollo solo) pensati per le reti domestiche, per questo fanno una serie nutrita di cose diverse tra cui collegare i dispositivi all'interno della stessa rete, ecc: si tratta di un'architettura distribuita basata su protocolli comunque standard come TCP/IP, HTTP, XML e SOAP (→ utilizzati normalmente per fare web service: SOAP come versione aggiornata di RPC sviluppato da SUN degli anni '80 il quale permetteva di effettuare chiamate a procedure remote tra macchine che condividevano la stessa rete; lo scambio tra un client/browser ed un server non è tanto diverso a livello di flusso di dati, così si sfrutta HTTP come veicolo per la richiesta remota al server per l'attivazione di "procedure remote").
Il problema che sorge con UPnP è che via web è difficile formattare per bene i parametri, per questo motivo ci si serve di XML che dal lato lessicale sfrutta la sua struttura a template standard e dal lato sintattico, una volta ottenuto il giusto file XML, SOAP si occupa di fare il *marshaling* dei parametri in modo da inserirli per intero all'interno di una stringa di caratteri nel file XML.
UPnP permette inoltre ai dispositivi facenti parte della stessa rete di rilevarsi reciprocamente senza doversi appoggiare ad un terzo elemento facente le veci di un server coordinatore degli accessi e delle interazioni (che all'interno di una rete domestica non è auspicabile). I dispositivi che supportano UPnP sono "plug and play", ovvero una volta collegati alla rete si dovrebbero configurare in maniera totalmente autonoma.
I dispositivi all'interno di UPnP si dividono in **device** (dispositivi passivi tra cui il UPnP MediaServer ed il UPnP Media Renderer) e **controller** (punti di controllo per le interazioni con e tra i vari device, tra cui il MediaServerCP, il MediaRendererCP ed il UPnP Control Point).



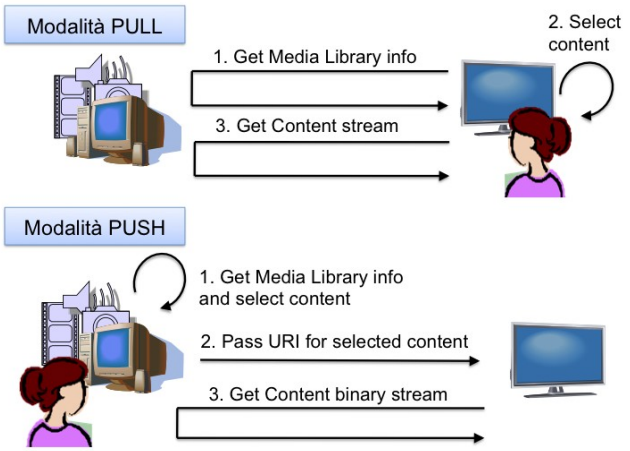
L'architettura di UPnP prevede uno stack protocollare ad esso dedicato il quale si appoggia interamente su IP (ecco perchè DLNA aveva bisogno di lavorare su Ipv4) perchè costituisce la tecnologia di rete più a basso costo che permette sia broadcast che multicast. UPnP ha poi una serie di protocolli ulteriori (poiché è una suite) di alto livello i quali si sono "scavati" una versione UDP di HTTP (anche se HTTP sarebbe di norma TCP bisogna ricordare che di HTTP era stata sviluppata anche una versione UDP, mai usata salvo che in questo caso) in modo da sfruttarla per lo streaming tramite incapsulamento su HTTP per passare le informazioni multimediali da server e renderer.



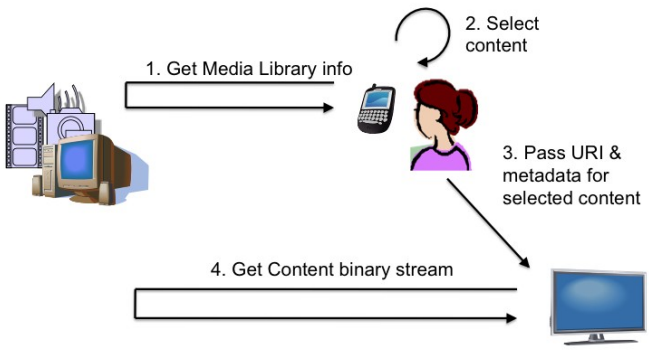
L'interazione tra device e controller avviene tramite sistemi di notifica in XML (i cui file contengono una descrizione esaustiva delle funzionalità di tutti i device e del controller; il file allora rimbalza continuamente tra i vari elementi). Il controller lo traduce poi in informazioni utili per capire quali tipi di file il device può supportare (da sostituirsi volendo alla sintassi di RDF per i dispositivi mobili). Le richieste di interazione sono inoltrate tramite SOAP e le notifiche asincrone viaggiano su GENA.
Il mercato odierno non solo non richiede, nemmeno sostiene questo tipo di tecnologia, che quindi continua a non trovare un'applicazione reale (chi al di fuori dei tecnici di settore vorrebbe un solo tasto per sincronizzare tutta casa?).
Ci sono diversi tipi di interazione tra i vari controller e device; lo scopo è riuscire

a facilitare la comunicazione dei dispositivi ignorando completamente la loro architettura interna e costruendo un'infrastruttura che li accetti tutti e li tratti tutti alla stessa maniera, in questo caso deve funzionare indipendentemente dai dispositivi che si troveranno nella rete (dev'essere quindi molto flessibile). Le principali classi d'interazione (entrambe discriminano a seconda di dove si trova l'utente rispetto al sistema) sono:

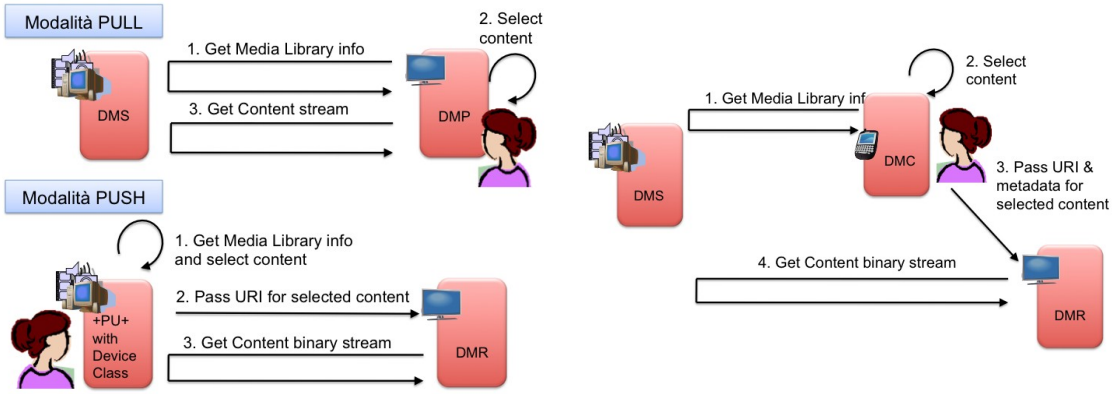
1. **Interazione "2 box".** Modalità PULL → l'utente si trova presso il dispositivo di fruizione: i contenuti si trovano salvati da qualche parte e si possono consultare dal dispositivo di fruizione, quindi dopo averne scelto uno si passerà a tale dispositivo l'indirizzo del contenuto il quale si occuperà prima di reperirlo chiedendolo al server di riferimento e quindi di eseguirlo. Modalità PUSH → l'utente si trova presso lo storage dei contenuti: come sopra salvo per il fatto che il renderer potrebbe essere più di uno così l'operatore sceglie quale contenuto visualizzare e su quale dispositivo farlo (anche su più dispositivi contemporaneamente);



2. **Interazione "3 box".** L'utente non si trova in corrispondenza né dei contenuti tantomeno del renderer, potrebbe essere ovunque e potrebbe non poter/voler comunicare con il dispositivo di fruizione per questioni fisiche o per impedimenti di varia natura, così si sceglie il contenuto da visualizzare tramite un secondo dispositivo (es: smartphone o tablet) e la prima parte del procedimento di scelta avviene come nell'interazione "2 box" di tipo PULL mentre la seconda parte del procedimento entro la quale avviene la fruizione del contenuto avviene come nell'interazione "2 box" di tipo PUSH.



In entrambi i casi potrei comunque avere due stream da coordinare tra loro. Posso comunque sostituire, negli schemi precedenti, il dispositivo con le classi di dispositivo ottenendo comunque lo stesso risultato.



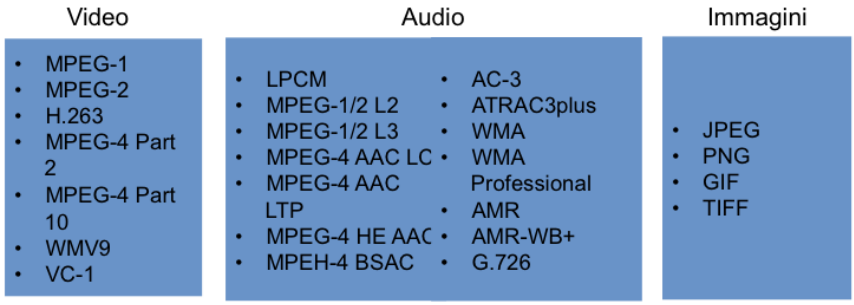
Nell'interazione "3 box" ci si potrebbe ritrovare meglio a livello di terminologia: il player lo si trova al posto del server solo se le risorse vanno cercate al di fuori della rete domestica (come nel caso di una trasmissione satellitare dove il player è il decoder).

Media management (UPnP AV 1.0). La suite UPnP possiede diverse sfaccettature a seconda dell'utilizzo che se ne deve fare: UPnP possiede una parte dedicata all'effettuare streaming effettivo, anche se il taglio dalla funzionalità prima descritta non è molto netto. UPnP è colui che rende possibile l'interazione del sistema con se stesso definendo una modalità di interazione tra i dispositivi di controllo e tutti gli altri; costituisce tuttavia una parte distante e scorporata che non ha nulla a che vedere con il trasferimento dei dati anche a livello protocollare (quindi è indipendente dal codec per quanto riguarda il formato dei dati); per questo il flusso video AV deve rimanere attivo anche quando il controller smette di interfacciarsi con il dispositivo. Di fatto si tratta delle funzioni utilizzate per implementare il modello di comunicazione visto sin'ora.
È possibile dividere tutte le possibili operazioni in base alla loro funzione:

1. **AVT** → AV Transport Service: controlla il flusso dei contenuti e permette di selezionare cose come la qualità di fruizione o di imporre la messa in pausa sulla visualizzazione;
2. **CM** → Connection Manager Service: controlla come i contenuti possono essere trasmessi per sapere quali trasmissioni sono effettivamente off/on line e quali il renderer può agevolmente fruire;
3. **CDS** → Content Directory Service: elenca i contenuti disponibili presso il server o scarica il palinsesto delle trasmissioni (nel caso satellitare);
4. **RCS** → Rendering Control Service: controlla come i contenuti sono riprodotti (in bianco e nero oppure a colori, volume alto o basso, ecc).

Media Format. Si sono fatte delle scelte basandosi su standard di mercato, così si sfruttano codec il più possibilmente aperti (stesso discorso per il decodec gratuito quando però i codec sono proprietari ed a pagamento). Si sono istituiti quindi dei *profili* relativi a classe dei media, categoria del dispositivo e regione geografica di utilizzo/provenienza. Con il termine **profilo** s'intende una combinazione di codifiche che permettono di ottenere un oggetto multimediale.

La lista riportata non è esaustiva ma evidenzia i codec più utilizzati, comuni e diffusi. Per l'audio ci sono problematiche maggiori rispetto al video, poiché è più soggetto alla pirateria e cambiano di molto i modi con cui su fruisce del contenuto.



I profili individuano anche tutti i codec gestibili sotto una stessa "etichetta" (che viene poi fisicamente apposta sull'apparecchio certificato). Sono stati definiti dal consorzio più di 350 profili ed un device multimediale può aderire a più profili contemporaneamente. L'etichetta apposta sul renderer indica i codec supportati, se il contenuto non ricade in uno di questi codedc è compito del player codificare opportunamente il contenuto. Il profilo aiuta anche a selezionare i contenuti disponibili su un media center in modo tale da scremare quelli che a livello di codec non sono supportati. Per ottenere una certificazione non è obbligatorio supportare TUTTI i codec ma solo un sottoinsieme, una volta ottenuta il device può mostrare l'etichetta del profilo: il server dev'essere in grado comunque di inviare contenuti codificati in almeno uno dei codec supportati dalla certificazione mentre i ricevitori devono essere in grado di decodificare TUTTI i profili della certificazione.

Content Distribution Network (CDN)

È un po' l'equivalente del DLNA ma questa volta a livello mondiale.
Si pensi di avere un server contenente certi dati molto gettonati che tutti vogliono scaricare (es: Youtube) → non è accettabile che esista un solo ed unico server contenente questi file al quale tutti si connettono contemporaneamente, specialmente se il servizio è globale e se il contenuto è molto voluminoso (come lo sono appunto i dati multimediali). Si può risolvere il problema moltiplicando il contenuto su più server sparpagliati per internet (a patto di controllare che i contenuti siano sempre sincroni a livello di aggiornamento su tutti i server ed in tutte le copie presenti, un lavoraccio), allora l'utente prenderà il contenuto dal server più vicino ed a lui più conveniente (anche a livello di costi di servizio). Un altro problema è ottenere una capacità ed un tempo di fruizione all'altezza del compito, che comunque rimangono sempre troppo bassi.
L'infrastruttura necessaria per replicare contenuti e reperire il "server migliore" prende il nome di **Content Delivery/Distribution Network (CDN)**. Dal punto di vista organizzativo ne esistono di due tipi a seconda delle disponibilità economiche di chi richiede il servizio e del traffico che ha intenzione di farci (oltre alle entrate che ha intenzione di collezionare): possono essere pubblici (allora verranno affittati dai gestori di servizio e la gestione dei server viene delegata all'affittuario, es: Google) oppure privati (quindi acquistati, mantenuti ed installati dalle aziende, allora chi distribuisce i contenuti possiede fisicamente l'infrastruttura, es: Akamai). Comunque sia, il funzionamento

tecnico è identico per entrambe le soluzioni.

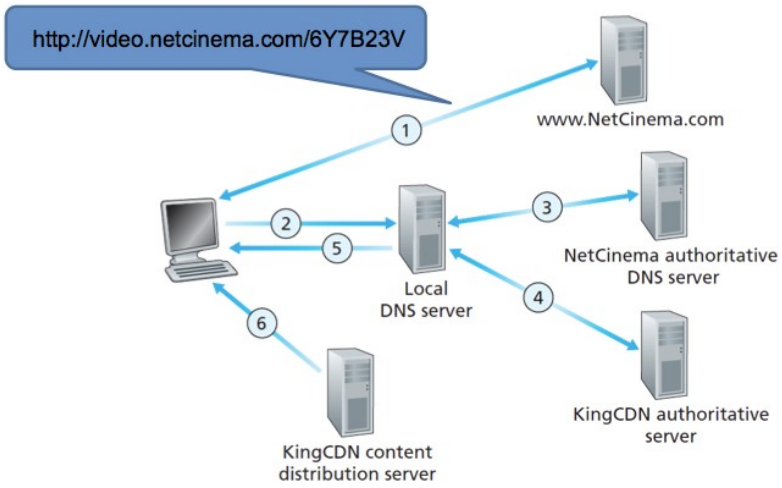
Akamai. È forse il più grande CDN provider esistente (da non confondere con il CDN più grande, attenzione: quello è Google) possedendo più di 27000 server sparsi per il mondo. È un'azienda che si occupa unicamente di ospitare e re-distribuire contenuti di terze parti. Ovviamente chi usufruisce del servizio ha a disposizione dei tool analitici per controllare i dati di traffico e per stimare quanto il servizio frutti a livello di entrate.

Esistono due filosofie di deployment differenti per le CDN:

1. **Enter deep** (Akamai): installare piccoli cluster nella rete d'accesso di molti ISP più grandi (ad alta velocità) così da minimizzare la latenza per una riduzione significativa del traffico (soprattutto per l'ISP), tuttavia è una scelta molto costosa. I server sono totalmente regalati ai gestori di backbone che ne possono usufruire come credono, nel mentre i CDN li affittano ai loro clienti sfruttando il fatto che si trovino già all'interno di una rete specifica (così da minimizzare il traffico esterno alla rete, abbattendo i costi sia dell'ISP che delle aziende clienti). Le CDN influenzano molto la scelta dell'apertura di un dominio.
2. **Bring home** (Google): installare grossi cluster in pochi punti strategici e collegarli con una rete privata ad altissima velocità, si hanno i propri server in corrispondenza dei punti di exchange delle grosse reti di service provider (i quali si connettono con POP e gateway alle dorsali) con sale collegate ai punti di controllo tra i vari servizi internazionali. Google ha fatto così; il provider non ci guadagna un gran chè, il traffico è tutto in uscita e non percorre lunghe distanze, tuttavia il costo è alto ma si ha una banda allargata al massimo dovuta al fatto che ci si trova in un punto di smistamento di dati. La rete viene fornita ai server in contratto d'uso, non viene costruita da zero (per abbattere i costi). Costituisce un'ottima soluzione se si hanno molti soldi da investire: si massimizza la banda tra i cluster, il traffico dell'ISP rimane sostanzialmente inalterato ed è comunque più economico su larga scala.

Una risorsa di qualsiasi natura custodita all'interno del CDN viene solitamente distribuita via HTTP. Il suo URL, tuttavia, non assomiglia per nulla a quello del sito che lo ospita, quindi da un URL diretto del tipo *www.sito.com/risorsa.jpg* si passa ad un URL del tipo *www.cdn.com/www.sito.com/risorsa.jpg*. In pratica il link deve poter cambiare in maniera dinamica la sua scrittura e la sua forma mantenendo comunque sempre funzionante il collegamento con la risorsa a seconda delle richieste del cliente (es: se il cliente è inserito all'interno di una rete Fastweb si dovrà avere un link Fastweb). L'URL del contenuto viene ri-mappato sul provider del CDN che restituisce una nuova alberatura appropriata (all'interno della quale trova posto l'URL originale). Si tratta di un meccanismo automatico utilizzato in maniera sistematica così da non dover per forza fare cambiamenti selettivi e manuali. Sostanzialmente il CDN si crea una mappa delle distanze tra gli ISP (o gli Autonomous System degli ISP) ed i suoi server, allora a seconda delle risorse richieste che pervengono gli utenti sono indirizzati al server a loro più vicino ed accessibile. Quando una richiesta arriva al Domain Name System del server autoritativo del CDN provider questo risponde in base all'ISP di provenienza. Per risolvere il nuovo URL si fa una chiamata al CDN creando una socket, quindi il server risponderà con l'indirizzo IP della risorsa cercata, possibilmente all'interno del service provider di

residenza del cliente e puntando al server ad esso più vicino. Se il trucco del DNS viene implementato direttamente sul sito proprietario del link (nel nostro caso *www.sito.com*) si ottiene lo stesso risultato senza utilizzare la redirezione (così fa Youtube usando la CDN di Google).



Posso aver a che fare con il link di una risorsa trovata su una certa pagina (chi si appoggia ad un CDN possiede contenuti con indirizzi diversi da quelli del proprio sito). Se si clicca sul link parte la richiesta ma prima il link va convertito: ci si riferisce al server DNS locale dove si scopre che l'indirizzo possiede un server autoritativo dedicato ai link provenienti da quel tale sito. Tuttavia il link non è esatto perchè non è proprio del sito ma del CDN, il DNS deve cercare il server autoritativo del CDN relativo a quella precisa risorsa: quando lo trova il server risolve l'URL e restituisce l'indirizzo alla risorsa che risiede, appunto, nel CDN.
Il cliente non seleziona da solo il cluster presso il quale la risorsa si trova (né sarebbe compito suo farlo), ma se ne occupa il CDN con molte difficoltà perchè non si sa mai a quale server bisogna rivolgersi (dubbio dovuto al fatto che la metrica di selezione cambia in continuazione in base alla distanza dei server, al traffico della rete ed alla presenza o meno della risorsa sui server); va da sé che la selezione del cluster basandosi sull'indirizzo IP della query DNS potrebbe non essere ottimale per una svariata serie di motivi:

1. il server/cluster geograficamente più vicino potrebbe non essere il più conveniente (per motivi di congestione della rete o per il fatto che potrebbe essere quello con meno risorse di tutti);
2. alcuni utenti potrebbero essere obbligati a servirsi di DNS locali molto al di fuori della loro rete d'accesso;
3. si potrebbero avere dei problemi per gli utenti mobili a seconda della loro posizione e del metodo di connessione (connettività) alla risorsa.

La selezione risulta essere migliore se si utilizzano dei metodi e dei meccanismi non basati sul DNS così da “misurare” la rete che esiste tra i cluster per capire quali tratti sono meno congestionati e quindi disponibili. La metrica utilizzata a riguardo potrebbe essere comunque sbagliata a causa di un calcolo fallito e sbagliarsi è preventivato (ed a volte sperato), in questa maniera è attraverso l'errore che si misura l'efficacia della soluzione corretta e maggiormente avvalorata (se si scopre invece che la scelta dovuta all'errore è più proficua si tende ad “aggiustare il tiro”). Ci si può affidare ad una scelta di *anycast*: dove

anycast è una specie di *presa in giro* delle tabelle di routing, se possiedo un certo indirizzo voglio segnalarlo a tutti i miei router comunicanti. Non contento i router lo comunicheranno a loro volta ai loro router comunicanti e così via dilagando nella rete. Ad un certo punto un router da qualche parte riceverà due (o più) segnalazioni per lo stesso indirizzo così da poter scegliere il percorso più breve tra quelli di cui è venuto a conoscenza (basandosi sulle metriche di routing definite per quella rete e per quello specifico nodo). La stessa cosa si può fare con le risorse di un CDN. Il rischio è che le tabelle di routing cambino durante la sessione (che può essere molto lunga); questo vuol dire che la socket venutasi a creare inizia a vagare senza meta senza spezzare la sessione quindi senza inviare una nuova richiesta per un nuovo invio di una richiesta per la risorsa, così a livello anycast una situazione del genere non è più gestibile (salvo che nel caso di attesa del timeout per il rinvio di una nuova richiesta alla risorsa).

Capitolo 8
Cluster e Data Center

Server Farm

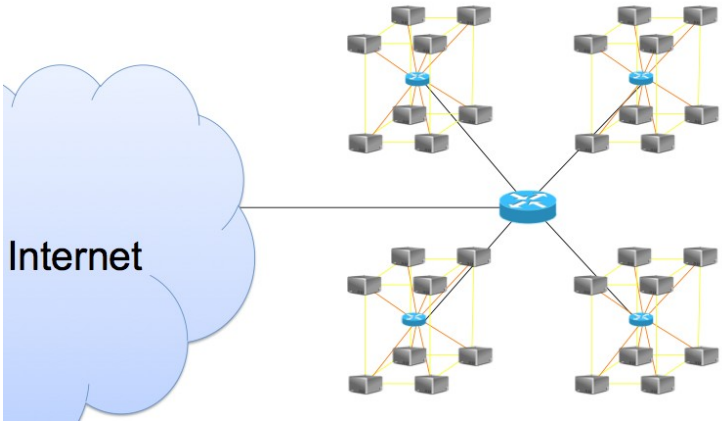
Innanzitutto non è affatto vero che quando si invoca l'indirizzo di una risorsa si colloquia sempre e comunque con un server. Se si blocca il server per qualsiasi motivo allora si blocca anche il servizio? E se si connettono in troppi? I server ci sono, ma non sono mai da soli. Le **server farm** non sono altro che architetture utili per erogare servizi (e non si parla solo di servizi web): a volte un solo server non è sufficiente per offrire un buon servizio a tutti gli utenti non riuscendo a gestire tutti gli accessi (con conseguente dilatazione del servizio la maggior parte di volte insopportabile). Una server farm riceve le richieste di servizio e prima di passarle ai server di cui è composta le fa "vidimare" e classificare da un apparato di rete (il quale risponde all'indirizzo IP invocato dal cliente) che sceglie a quale server passare il testimone. È vero che l'apparato di rete potrebbe sovraccaricarsi ma è altamente improbabile perchè succederebbe solo se fosse sottoposto ad una quantità di invocazioni drasticamente superiore a quella che guasterebbe il servizio di un server qualsiasi.
I vantaggi di una server farm si individuano nel:

1. load balancing: si mandano le richieste ad una macchina drasticamente meno carica ed è possibile sopportare carichi complessivi molto elevati;
2. fault tollerance: se uno degli host si guasta basta escluderlo dal servizio;
3. scalabilità: se aumenta il numero di utenti basta aumentare il numero dei server e non si deve intervenire per modificare e cambiare il servizio "d'interfaccia";
4. trasparenza: l'utente finale ignora completamente che si sta collegando ad una nutrita quantità di server.

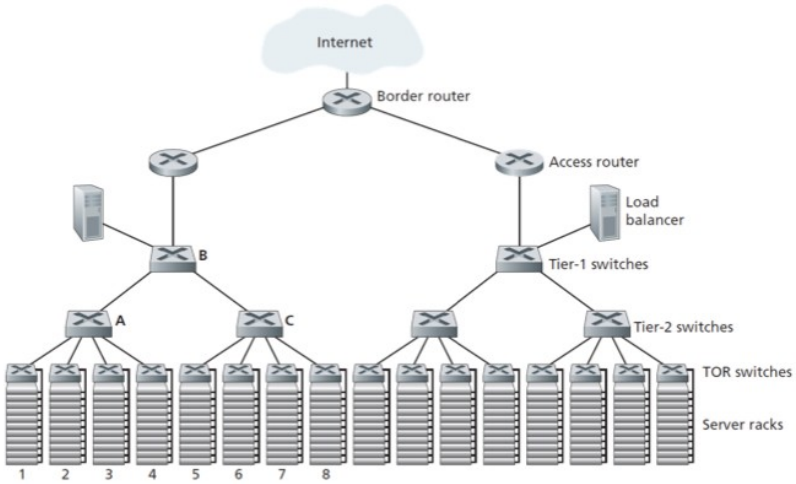
Cluster

Nonostante la risoluzione apparentemente proficua a livello di rete, i calcolatori sono ancora soggetti a guasti che possono invalidare qualsiasi servizio.
I **cluster** sono composizioni di 2 o più calcolatori messi in relazione hardware molto stretta (solitamente collegati da cavi) per riuscire a monitorarsi a vicenda: solitamente una delle due macchine lavora e l'altra rimane inerte a controllarla, quando la macchina lavoratrice si guasta quella inerte se ne accorge e manda l'allarme di guasto (la seconda macchina è dedicata a questo scopo perchè una macchina che di solito non fa nulla è più difficile che si guasti) per poi prendere le fila del lavoro lasciato a metà dall'altra macchina.
Dal punto di vista funzionale si tratta di un hardware che si comporta come se fosse una macchina sola: anche i cluster nel loro piccolo sono nati per bilanciare il carico computazionale condividendo tutte le loro risorse (come i dischi) e le applicazioni rispetto alle quali si vorrebbe fossero trasparenti (si vorrebbe che un servizio non sapesse su quale parte del cluster viene eseguito anche se oggi non è ancora possibile → comportamento di un sistema distribuito dove si ignora la dislocazione e la presenza stessa dei nodi).
I cluster si utilizzano come componenti delle server farm tuttavia costituiscono un concetto separato dalla rete (i cluster sono i nodi che costituiscono la rete delle server farm). Lavorare con i cluster semplifica la vita: quando arriva la

richiesta per una risorsa è il middleware fornito con i cluster ad occuparsene. Solitamente le strutture dei cluster sono dette ad *ipercubo* nel senso che si estendono sempre in potenza di 2.



Fare routing all'interno di un cluster, però, è alquanto difficoltoso: è importante definire delle regole precise di routing interno (per gestire casi in cui, per esempio, la richiesta viene indirizzata ad un nodo anche se la risorsa sta su un altro). I sistemi di clustering si concentrano tipicamente su un sistema di calcolo o su un servizio; una struttura IT tipicamente deve avere molti di questi cluster inter-operanti tra loro.
Ma quanto spazio serve per organizzare una server farm e quanta energia consumano tutte quelle macchine? È per ovviare a questa esigenza che nascono i **data center** che sono luoghi fisici i quali sono pensati apposta per mettere a disposizione le risorse necessarie per l'alimentazione e la gestione di tutte quelle macchine; costituiscono la globalità dell'infrastruttura fisica di cluster collocata in un luogo geografico. Le infrastrutture ospitate nei data center sono costituite da macchine secondo il format industriale dalla forma molto sottile le quali vanno poi inserite in armadi blindati già predisposti a livello di collegamenti per il clustering i quali forniscono infine l'energia per alimentare le macchine senza l'ingombro ed il calore dell'alimentatore interno. Solitamente nei data center l'aria condizionata è tenuta a 10° costanti tutto l'anno. Il 60% dei costi di mantenimento sono costituiti dall'hardware che va sostituito con frequenza di 3-4 anni (per questo si tende ad acquistare hardware di fascia bassa con Linux come sistema operativo); il 15% sono spese energetiche.



Non è vero che i data center sono tutti come quello di Google: esistono diverse varianti sul tema a seconda dell'uso e della forma delle macchine. Persino la nomenclatura cambia a seconda di chi produce l'hardware e fornisce i servizi.
Come da figura, ci sono tanti **rack** (armadi di computer industriali così configurati per ridurre il volume di spazio occupato dalle macchine) serializzati ciascuno dei quali ospita una quantità di macchine in cluster: un rack costituisce quindi un cluster. In cima ai rack si trova il **TOR Switch** (collocato in cima così da far ciondolare i cavi lungo l'armadio, in questa maniera è più facile ed immediato trovare dove un cavo si è eventualmente scollegato semplicemente perchè penzolerebbe o se ne starebbe sciolto in terra) che costituisce la porta di comunicazione del rack con la rete (e quindi anche con gli altri rack). Si ha quindi una tipica gerarchia cluster con i vari **tier switch** come nodi della rete. Il **border router** costituisce l'uscio di comunicazione con l'intera internet.
Come variante di questo schema (che è il più semplice) si può pensare ad una maglia completa nella quale la gerarchia si complica per offrire però percorsi interni più scarichi e liberi dal traffico (ma quanto cavo mi serve??); non si può far collassare tutto su un unica macchina perchè se si guasta s'invalida tutto il servizio (oltre al fatto che spesso questi cluster sono geograficamente sparsi in maniera poco omogenea); per evitare il *tranciamento* dell'anima metallica in rame di qualsiasi cavo (che può avvenire per qualsiasi motivo ed in qualsiasi momento) se ne mettono in abbondanza così nel caso di un guasto non c'è bisogno di intervenire direttamente (anche perchè è difficilissimo alle volte capire dove il guasto è avvenuto) senza troppe complicazioni.
In alternativa al clustering gerarchico si può optare per altre soluzioni:

1. mettendo tutti i calcolatori in cerchio per ridurre i tempi di accesso alle risorse, nel caso pessimo, almeno della metà come pensato dalla CRAY durante in progetto ASCII (1998) per la decifratura di frammenti di codice in maniera automatica (allora emerge il problema di quale strada prendere per accedervi, se a destra o a sinistra);
2. come fa Google, organizzare dei *container* (come quelli per il trasporto navale delle merci) pieni zeppi di hardware all'interno dei quale nessuno entra e che fungono da “moduli” ai quale ci si interfaccia con dei connettori esterni. I moduli vengono utilizzati fino al completo guasto delle loro componenti (il quale viene monitorato ed accertato sino ad una certa soglia), momento in cui vengono distrutti e ricostituiti;
3. collocare tutte le macchine all'interno di un vulcano spento al di sotto di un ghiacciaio perenne così da risparmiare sul sistema per il raffreddamento come fa un'azienda di videogame islandese (CCP Games famosa per EVE Online) sfruttando così la conformazione geografica del territorio e la grandissima connettività che solo paesi come l'Islanda e pochi altri possiedono (per una questione di isolamento che porta la popolazione ad un'esigenza di connessione molto alta).

Cloud Computing

Il *cloud computing*, che non va assolutamente confuso con il cluster, costituisce un paradigma ottimizzato di virtualizzazione delle risorse e distribuzione di carico computazionale. Tipicamente si finisce col costruire un'infrastruttura cloud appoggiandosi direttamente su un data center (quindi ad una gerarchia di cluster) ma rendendo il tutto trasparente all'uso ed all'allocazione delle risorse.

Capitolo 9
Adaptive Multimedia

Multimedia adattivi

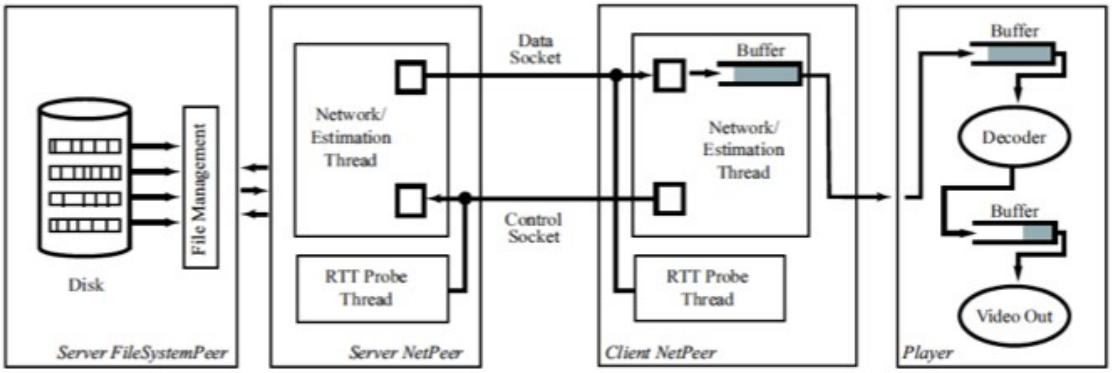
Si può classificare come **adaptive multimedia** qualsiasi tipo di trasmissione multimediale che si adatta alle condizioni di contorno per poter garantire all'utente una buona fruizione del contenuto. Non ci è dato di sapere con grande certezza cosa s'intenda per *condizioni di contorno*: possono essere la rete più o meno congestionata che per tutta risposta sopporta una quantità più o meno ampia di dati inviati su di essa; il contesto situazionale nel quale si trova il client così da giustificare la fornitura di dati in una certa maniera piuttosto che in un'altra; il dato che cambia la sua natura intrinseca a seconda della richiesta dell'utente e coerentemente con il contesto (il che però solleva un problema: dipende tutto anche dalla natura del contenuto e dalle caratteristiche che lo rendono caratteristico e che quindi non possono essere stravolte e sacrificate). Si tratta comunque di situazioni all'interno delle quali spesso non erogare alcun servizio è assai meno lesivo dell'immagine dell'azienda piuttosto di erogarlo comunque ma in una maniera ed in una qualità così poco ottimale da fornire un servizio degradato che non piace per nulla all'utente (contrariandolo non poco soprattutto nel momento in cui magari paga per poter usufruire del servizio).
In questa maniera si rinuncia quasi definitivamente al fornire una qualsiasi qualità di servizio: l'adattamento di un contenuto multimediale va inteso come una tecnologia da sfruttare nel momento in cui non è possibile garantire nessun tipo di qualità di servizio (perchè comunque, per sua stessa definizione, ignora qualsiasi tipo di QoS a favore di una trasmissione a tutti costi di contenuti spesso nemmeno fruibili). Anziché garantire la disponibilità delle risorse (visto che si adatta il contenuto è ovvio che le risorse per poterne fruire non sono disponibili) non si garantisce nulla di preciso e si definisce unicamente uno SLA in termini percettivi e indipendenti dalle risorse (che per sua natura non è mai troppo preciso): importante e fondamentale da fare perchè esistono situazioni sintomatiche per le quali la QoS non è assolutamente garantita e non si è in grado neppure volendolo di garantire le risorse richieste (e guarda caso sono le situazioni tipiche per le quali il mercato moderno tende a spingere chiedendo contributi monetari ai clienti, come la fruizione di contenuti da dispositivi mobili, connettività wireless e reti cellulari). Il contenuto va quindi tarato in base alla disponibilità di risorse spesso cambiando i parametri della codifica o la natura del contenuto stesso (che è modificabile). Quando lo stato della rete peggiora lo stream se ne deve accorgere e modificare di conseguenza la codifica della trasmissione sacrificando quel poco che serve della qualità del contenuto per salvaguardarne però la corretta trasmissione.


Bisogna capire rispetto a cosa va effettuato l'adattamento:

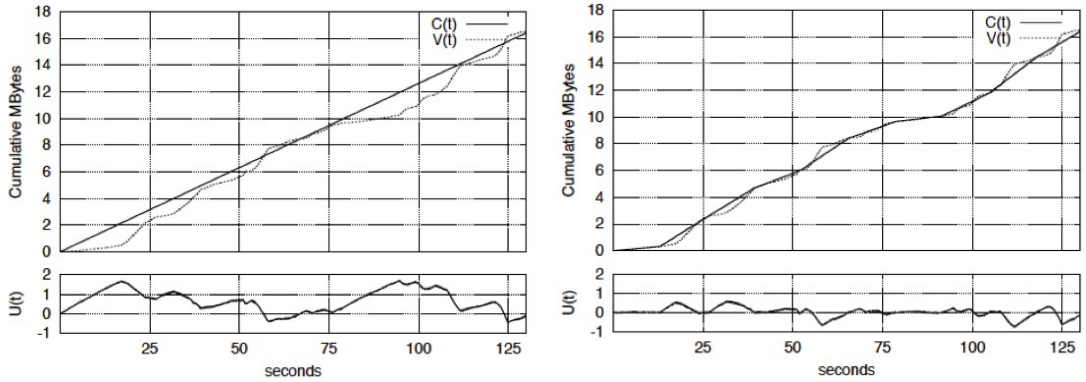
1. a seconda delle condizioni della rete: si deve verificare che la banda sia sufficiente oppure no, oppure che l'end-to-end delay non sia significativo (cosa che vale solo per il discorso della comunicazione simil-VoIP come nel caso di Skype ma non nel caso dello streaming video di Youtube). Di norma, se il ritardo è troppo alto si ferma la trasmissione e si verificano le condizioni della rete, magari caricando il buffer di arrivo, così poi da ripartire con la trasmissione appena le condizioni si sono normalizzate;
2. ai dispositivi: ed al passaggio tra un dispositivo e l'altro. Esistono casi in cui ha senso migrare da un dispositivo più limitato ad uno più esteso ma non sempre questa necessità si verifica davvero (c'è davvero richiesta di un servizio simile?). Di norma ogni utente si collega con il dispositivo di cui è fornito: tempo fa esistevano tantissime codifiche diverse per venire in contro alla moltitudine di dispositivi portatili differenti, oggi (poiché il collegamento mobile può essere solo di tipo broadband e cellulare) si tende a definire due codifiche principali (una in alta e l'altra in bassa qualità a seconda della disponibilità di banda) e poi a seconda del formato del dispositivo le trasmissioni verranno adattate di conseguenza (che è ciò che fa Youtube in maniera totalmente automatica anche senza un metadato che sfoltisca le richieste, tuttavia il carico di lavoro sembra non essere troppo oneroso);
3. alle situazione in cui si trova l'utente: per esempio se è in giro a piedi oppure in macchina, allora la fruizione dei contenuti dovrebbe adattarsi al contesto anche in termini di contenuti erogati (cosa che si trova raramente in commercio e quei pochi prodotti che lo fanno si basano sulla tecnologia hardware per farlo in maniera un po' bieca e non molto pulita).

Adattarsi alla rete è ciò di cui si è largamente discusso parlando di architetture software: si ci basa su algoritmi molto semplici i quali innanzitutto fanno una stima delle risorse di rete a disposizione, mandano poi una richiesta al server conforme alle disponibilità ed il server risponderà allora con la codifica opportuna perchè possa essere fruita dalla situazione di rete corrente. Esistono però dei difetti sostanziali in questo approccio riguardanti l'affidabilità della stima fatta dipendente dalla natura del contenuto richiesto, senza parlare della reattività della comunicazione (magari la situazione di congestione si risolve prima che il contenuto venga adattato per essere erogato ad una qualità inferiore senza poter revocare la richiesta e nemmeno richiedere di alzare nuovamente la codifica). La **stima delle risorse** lascia sempre un po' a desiderare: non si è mai davvero in grado di capire quando le risorse mancano e nemmeno quando se ne hanno più del necessario. Purtroppo **i server non sono standardizzati** e non esistono di conseguenza metodi considerati standard per segnalare al server le condizioni della rete, spesso il tutto è legato alle implementazioni proprietarie delle specifiche ed il tipo di segnalazione varia in base al contenuto ed alle modalità di fruizione. Inoltre non si capisce mai **quando è il momento giusto**: l'adattamento, al di là delle limitazioni tecniche, dev'essere fatto quando è realmente richiesto e realmente necessario: non va bene correre ai ripari troppo presto perchè si potrebbe decrementare la qualità della fruizione del contenuto a fronte di una congestione brevissima e temporanea, ma nemmeno farlo troppo tardi perchè chi può dire per quanto tempo l'utente deve “vedere e sentire male” prima che il sistema si adatti?? Per tentare di capire quando veramente inizio a perdere dei dati si costruiscono

delle euristiche per calcolare la soglia sotto la quale è necessario compiere l'adattamento. Si fanno anche delle stime percettive (che per l'orecchio umano stanno attorno ai 2 sec) per determinare quei limiti entro i quali la perdita delle informazioni può ancora risultare tollerabile.

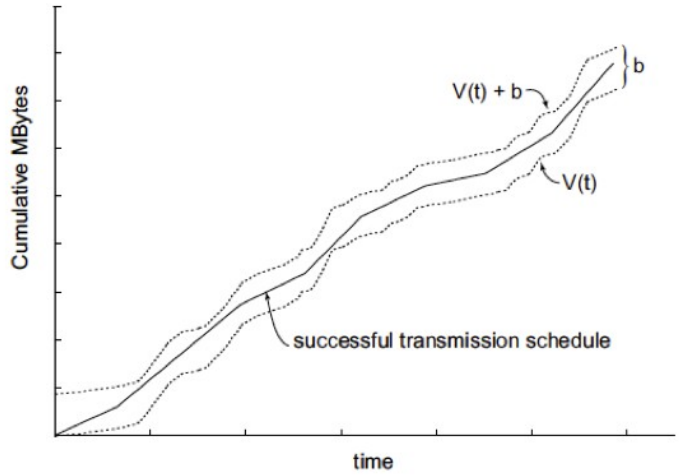


VTP Streamer Architecture. Si hanno quattro entità diverse. Nel Server File System Peer si dispone di una serie di codifiche multiple contrassegnate da dei marcatori temporali. Tra il Server NetPeer ed il Client NetPeer il file viaggia nella rete attraverso una socket dedicata al traffico dati ed una gemella dedicata al traffico di controllo. Se si utilizzasse solo la socket per il traffico dati sarebbe difficilissimo controllare la codifica giusta con la quale fruire del contenuto, così si manda un segnale di controllo in maniera temporizzata (stimando così la quantità di banda disponibile). Il client comunica allora con il server mediati anche da dei thread paralleli alle socket che rimangono in disparte in ascolto della comunicazione per compiere delle stime e delle misurazioni (così che se qualche cosa non quadra verrà fatto notare alle socket e di conseguenza si aggiusterà il tiro con una codifica diversa): l'idea è quella di riuscire a dare un feedback della trasmissione.
Come dev'essere fatto il *playout schedule*? Se il buffer di destinazione è limitato lo schedule non può essere chiaramente lineare (anche se così si vorrebbe). Senza considerare tutto il contorno, solitamente la richiesta di buffer è data dalla differenza tra lo schedule reale e quello presunto (*CBR Playout Schedule*).

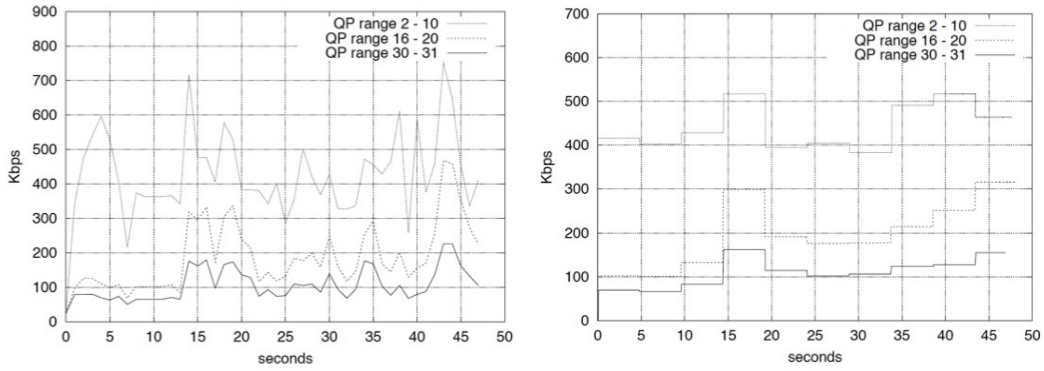


È visibile allora dallo schema della differenza tra i comportamenti di schedule (seconda parte del primo grafico) che esistono diversi casi di schedule underrun decisamente profondi (ed indesiderati). Per migliorare la situazione si prova a spezzare la trasmissione in segmenti (rappresentati dalla retta spezzata del secondo grafico): il tutto si complica per via dei parametri che andranno valutati con più attenzione, tuttavia la situazione, invece di migliorare, peggiora (nella

seconda parte del secondo schema la differenza tra i due schedule evidenzia una quantità maggiore di casi in cui si incorre nello schedule underrun). È possibile effettuare un'analisi a secco su qualsiasi video (o contenuto multimediale) già codificato per stimare quale possa essere il punto/picco più basso del suo grafico corrispondente al valore di prefetch del buffer di arrivo.

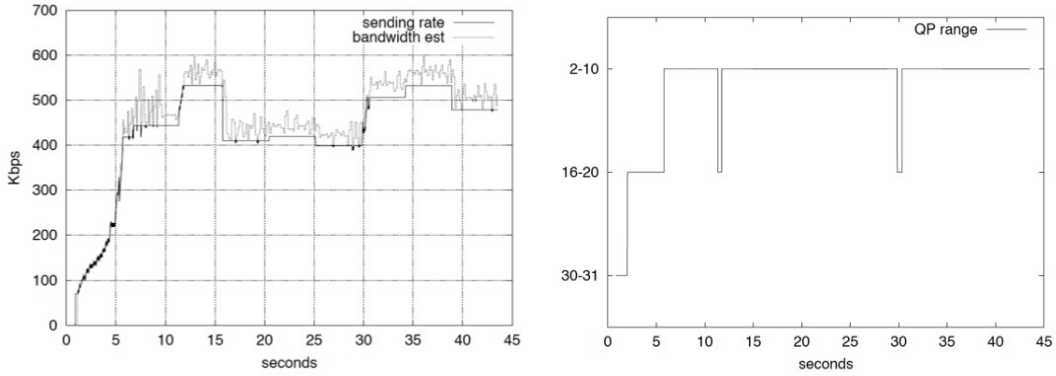


Si disegna allora un grafico a partire da due linee (quelle tratteggiate) le quali corrispondono la prima (quella più bassa) al valore di prefetch del buffer e la seconda alla prima alla quale si somma la dimensione reale del buffer. Il corridoio che si viene a formare tra le due linee corrisponde alla dimensione del buffer d'arrivo: la trasmissione non deve mai toccare le due linee, così si può pensare di spezzare il contenuto multimediale per tentare di non sforare mai dal corridoio disegnato. Questo significa che il problema è facilmente risolvibile mediante programmazione lineare (ovvero utilizzando formule matematiche di base per nulla complesse): a priori si sa già l'intervallo per il quale lo streaming è considerato davvero in tempo reale (*Admissible Streaming Solution*).



VBR Bandwidth Usage. Lo schema riporta la banda utilizzata da tre tipi di codifica differenti: tutte tagliano nello stesso punto e costituiscono il sistema da sfruttare per effettuare il salto da una codifica all'altra. Nel secondo grafico si nota che all'interno di un certo arco temporale una codifica rimane sempre la stessa. Per compiere il salto tra codifiche si è provato a mandare pacchetti con maggiore frequenza i quali controllavano se era possibile accedere alla codifica superiore essendo eventualmente a disposizione le risorse necessarie, peccato che come sistema non funzionasse troppo bene. Allora si è pensato di partire sempre dalla codifica minima ed, utilizzando il metodo di TCP, ad ogni Round

Trip Time di allargare la finestra di congestione inviando un pacchetto in più: se così facendo si riesce ad intercettare la banda al livello superiore il salto può essere effettuato (e si fa). Il risultato a livello di qualità percettiva è dato dal grafico a gradini (il quarto grafico) ed è complessivamente buona: si degenera solo in pochissimi punti ed il profilo mantenuto è sempre quello della codifica più favorevole.

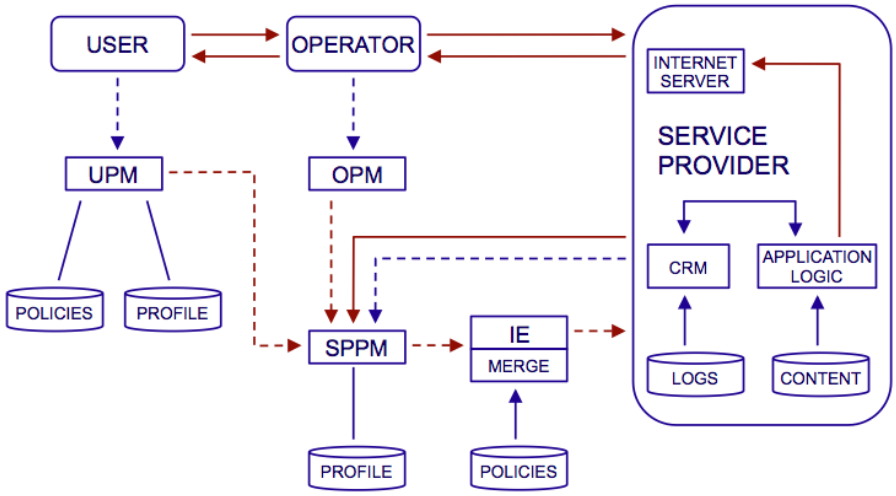


Adattarsi ai dispositivi

Quando si tratta di considerare la scalabilità verso il basso (quindi da una qualità maggiore ad una decisamente minore) l'adattabilità ai dispositivi non è mai un problema troppo complicato (anche se la scalabilità verso il basso vera e propria è una cosa diversa dall'adattabilità dei dispositivi). Solitamente se un dispositivo non riesce a sostenere una certa qualità di trasmissione lo segnala ed il server si comporta di conseguenza.
In questi ultimi anni persiste la moda di voler usufruire di multimedia sui cellulari e sarebbe bello identificare dei servizi veramente innovativi perchè il video "e basta" potrebbe non avere poi più molto senso. Il guaio è che i dispositivi moderni, soprattuto se mobili, non sono affatto omogenei avendo tutti caratteristiche hardware e software differenti (ma soprattutto hardware).
Il W3C ha sentito la necessità di istituire un nuovo standard chiamato quindi **Resource Description Framework (RDF)** il quale costituisce un metodo generale per la descrizione concettuale delle informazioni contenute in una risorsa web (ed anche i dispositivi mobili, quando sono collegati alla rete, possono essere visti come risorse web). Si tratta di un file XML nel quale un dispositivo descrive le sue caratteristiche tecniche attribuendosi l'appartenenza ad una determinata classe (RDF distingue le risorse web in classi come se facessero parte di una specifica tassonomia). Una volta ottenuti i parametri importanti (e standard) di un dispositivo registrati nel RDF li si inserisce in appositi spazi inutilizzati dell'header della richiesta HTTP: si suppone che sotto ci sia un'infrastruttura di comunicazione web che al giorno d'oggi è quasi una facilitazione contrariamente ai tempi andati in cui sfruttare HTTP era più un danno che una conquista (anche per via dei firewall che non sempre lasciavano passare le richieste HTTP). Si manda poi la richiesta HTTP al server che legge i dati dell'RDF (se supportato, sennò semplicemente li scarta ignorandoli): qualsiasi server web allora potrà erogare contenuti personalizzati ed i parametri condivisi verranno utilizzati poi per pilotare lo streaming dal server.
RDF da solo non basta, è solo il punto di partenza perchè non fa altro che rendere noti alcuni parametri che comunque sono duraturi nel tempo e non subiscono modifiche nell'immediato: manca ancora tutta la parte

dell'adattamento realtime e per quello l'unica soluzione moderna è costituita dall'implementare delle infrastrutture ad hoc (e nessuno ha ancora sviluppato qualche cosa che vada bene per tutte le situazioni).

CARE. Si tratta di un progetto proposto da persone che lavorano nel dip. di informatica dell'Università degli Studi di Milano nel contesto di un progetto FIRB (dal 2002 al 2006).



Si hanno 3 elementi in gioco: l'utente, l'operatore ed il service provider. Per ciascuno di loro si definisce un profilo a seconda delle loro qualità e delle loro richieste/funzionalità; il service provider ha tuttavia un profilo suo ed una struttura particolari che non andrebbero mai toccati per nessun motivo. L'idea di fondo è quella di unire i tre profili all'interno di un motore (intelligenza artificiale) che in base a certe policy li fonde assieme per poter dare dei suggerimenti al service provider. I tre oggetti sono compartimenti stagni a sé stanti ed il motore lavora solo con i loro profili in maniera super partes ed anonima senza interessarsi degli eventuali dati sensibili (che magari possiede l'utente e che non ha interesse di condividere con chiunque sulla rete).
Realizzare questa architettura non è stato per nulla semplice e banale.

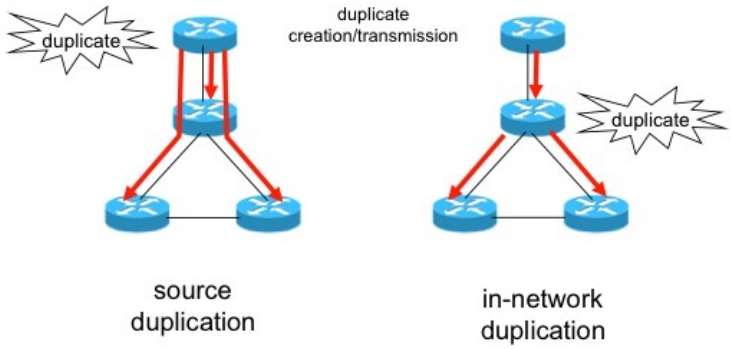
Capitolo 10
Distribuzione dati multicast

Multicast

Si tratta di tecniche per la distribuzione da un host a molti, cadono sotto questa definizione tutte le comunicazioni che si svolgono punto-a-punto e che non interessano nello specifico TUTTI i nodi della rete ma solo un sottoinsieme di questi che si possono definire i “nodi interessati”; si parla comunque di routing a livello 3 (che costituisce l'approccio più coerente). Ci si chiede se il multicast alla fine sia da considerarsi come una soluzione auspicabile: raggiungere tutti con un solo pacchetto che poi va moltiplicandosi è antieconomico, tuttavia il multicast è ancora molto praticato (es: la tv online di Fastweb che all'inizio non aveva sistemi di protezione e di account così da permetter a chiunque di connettersi e “sbirciare” qualche canale, ma è un'altra storia).
Il **multicast** fa una comunicazione da una sorgente a molti destinatari; non si tratta di una strategia di routing di per sé anzi ne sfrutta di già esistenti (e per l'instradamento predilige quelle tabelle di routing che contengono tutte le corrispondenze di più indirizzi IP possibili) per compiere un'instradamento che non è più un tradizionale da punto-a-punto; non garantisce nessuna qualità di servizio perchè tra le altre cose non differenzia i flussi e non è prettamente devota al multimediale perchè il multicast è indipendente dal contenuto trasportato. Il multicast come tecnologia però non si può però ignorare, perchè molto spesso le comunicazioni multimediali vengono trasmesse da una sorgente ad un insieme di destinazioni (come capita con i canali televisivi delle tv via rete). La documentazione relativa al multicast è abbastanza chiusa (e si salvano solo i documenti relativi a qualche sviluppo indipendente).
Ovviamente tra broadcast e multicast ci sono delle differenze, anche se non particolarmente sostanziali. È ovvio che non tutte le comunicazioni avvengono secondo un modello uno-a-uno: si parla di **broadcast** quando il contenuto è inviato ad ogni nodo presente nell'infrastruttura; la comunicazione broadcast si appoggia su una rete che non è forzatamente broadcast (basta che il pacchetto sia segnato broadcast e saranno i router a provvedere al corretto instradamento secondo il protocollo broadcast). Il fatto che il pacchetto venga mandato a TUTTI non è proprio corretto: sempre e comunque si fa in modo di determinare un sottoinsieme di nodi all'interno di una certa sottorete e per far ciò si isola un intervallo di indirizzi IP. Il **multicast** invece invia i suoi contenuti solo ad una lista di iscritti al servizio, per cui non si parla più di un range di indirizzi ma di vere e proprie liste all'interno delle quali un nodo che vuole entrare a far parte del multicast deve iscriversi. Nel multicast si vuole mandare un messaggio solo ad alcuni host: se si inviano informazioni riservate si vorrebbe che un punto intermedio non sia capace di decifrarne il contenuto.
L'instradamento broadcast consuma moltissime risorse. Bisogna ricordare che broadcast e multicast non si trovano in antitesi, anzi il multicast è un caso più specifico di broadcast. Se si hanno 3 destinatari di un certo contenuto ed una sola sorgente, si vorrebbe inviare tale contenuto a tutti i richiedenti e per far ciò esistono un paio di opzioni alternative:

1. si possono fare 3 copie della risorsa in maniera da creare 3 socket differenti ciascuna dedicata ad uno dei client richiedenti. In questa

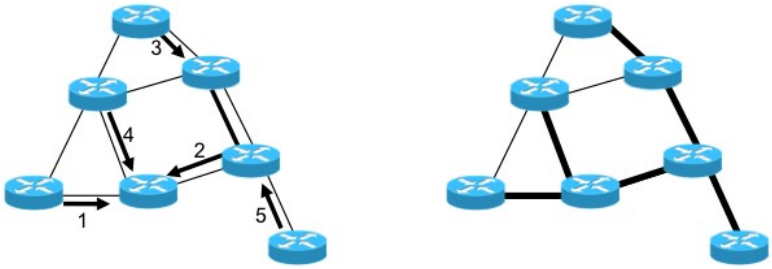
maniera si terrà occupata la banda del primo link del grafico che dovrà sostenere 3 flussi identici. Si può fare anche moltiplicando la sorgente (come fa Youtube);
2. si possono moltiplicare le risorse al momento della diramazione facendo così delle copie qualora ce ne sia una reale necessità.



La duplicazione dei pacchetti alla sorgente è inefficiente perchè si sprecano troppe risorse, molto meglio moltiplicare la risorsa durante il percorso. Ma come può fare la sorgente a sapere a priori l'indirizzo di tutti i destinatari? Qualsiasi router in questa situazione fa da portavoce per la più o meno piccola sottorete locale che gestisce (rete che si da sempre per scontata ma bisogna sapere che non è il router a "chiedere" qualcosa, piuttosto è uno, o più, degli host della sua sottorete). Non è detto che in qualsiasi situazione questo sia il sistema più economico perchè nella realtà ogni link tra router ha un *peso* che va preso in considerazione (allora il calcolo per determinare il tragitto ottimo va effettuato per ogni pacchetto in maniera automatica).
La duplicazione dei pacchetti è effettuata tramite:

1. **flooding (inondazione)**: ovvero un pacchetto contrassegnato come broadcast va inoltrato nella rete a tutti gli altri router comunicanti con l'interfaccia di passaggio (che ha il compito di duplicarlo prima di "spargerlo" per la rete) anche se si rischia di incorrere in casi di loop (dove infinite copie di quel dato pacchetto broadcast continuano a percorrere il ciclo all'infinito); se la rete è abbastanza piccola si può sfruttare il TTL (*time to live*) per impedire la formazione di cicli ma solo se si possono sprecare molte risorse;
2. **flooding controllato**: fa una leggera variazione sull'algoritmo precedente. Poiché ogni pacchetto conosce l'indirizzo della sorgente dalla quale è stato emanato (caratteristica di qualsiasi pacchetto IP) lo si duplica solamente se è arrivato da quell'interfaccia di rete (del router corrente) con la quale il nodo che lo riceve solitamente comunica con la sorgente. Il router che lo riceve allora lo inoltra a tutti gli altri router salvo a quello da cui il pacchetto è arrivato. In questa maniera è come replicare i dati solo verso "il basso" così da impedire il ritorno del flusso. Si parla anche di *Reverse Path Boradcast (RPB)*;
3. **spanning tree**: si può costruire un albero di copertura per eliminare tutti i pacchetti ridondanti. Per prima cosa (prima di inviare qualsiasi dato) si costruisce la struttura dell'albero di copertura; se si ha a che fare con una sorgente di traffico si avranno, nella direzione opposta a quella del traffico, dei messaggi di richiesta da parte degli altri router di potersi iscrivere alla lista del multicast. Questi messaggi di richiesta effettuano

un certo percorso (così si comporta anche RPB). Poiché ogni destinazione potenziale manda un messaggio verso la sorgente viene considerato come punto di biforcazione il primo router già facente parte dell'albero incontrato dal messaggio (quindi già conteggiato).
Lo spanning tree costituisce un insieme di archi sovrapposti a quelli del grafo che descrivono la rete da utilizzare per assicurarsi la corretta comunicazione tra i nodi in maniera tale da consumare meno risorse possibili. Operativamente uno spanning tree crea una diramazione ogni volta che viene mandato un messaggio e si aggancia al primo router che incontra e che risulta essere già agganciato allo spanning tree (quindi già collegato alla sorgente mediante il cammino più ottimo possibile). Esistono protocolli di rete che formalizzano all'interno delle tabelle di routing i dati degli spanning tree così da gestire agevolmente un qualsiasi pacchetto broadcast.



Il broadcast però non è sinonimo di multicast. Può capitare di essere costretti ad attraversare alcuni router che non fanno parte del gruppo di router interessati alla comunicazione multicast in corso (e che non possono/devono vedere il contenuto del pacchetto). Il sistema di consegna ottimizzato dagli spanning tree potrebbe non coincidere con l'albero effettivo per la consegna con il minor spreco di risorse (dipende anche dalla natura dell'informazione trasportata perchè se l'albero di distribuzione va lasciato in piedi per lunghi periodi vale anche la pena ottimizzare secondo le risorse e meno secondo la politica d'interesse). Generalmente non capita mai di ricadere in questa casistica e gli alberi sono sempre molto dinamici: si può quindi scegliere di utilizzare un albero molto granulare (come nel caso di Fastweb) così il multicast risulta essere fatto a “livello di quartiere” (nel caso della tv via web) altrimenti, come nel caso delle comunicazioni satellitari (SKY) scelgo solo a valle se un utente è abilitato a visualizzare il contenuto oppure no.
Un protocollo multicast deve saper gestire dei gruppi sapendo quindi chi può ricevere quali informazioni; deve saper costruire un albero di distribuzione per definire ed effettuare un routing intelligente (per consegnare le informazioni a coloro che ne hanno fatto richiesta e magari pagano per il servizio; ricordiamo che il multicast nasce per le trasmissione televisive quindi anche per la consegna di contenuti in realtime e multimediali) e per garantire la consegna dei dati secondo i parametri richiesti. Sull'ultimo punto ci si è un po' lasciati andare per il fatto che oggigiorno non c'è molta differenza tra ciò che per esempio si sta guardando a Milano ed a Roma (salvo qualche millisecondo di scarto che però non compromette la trasmissione). La *gestione dei gruppi* prevede la nozione di “gruppo multicast” e di iscrizione a questo gruppo: è il router d'accesso alla rete locale che si impegna a far richiesta d'iscrizione ed a disiscriversi se serve facendo da intermediario tra il servizio scelto ed uno (o più) degli host della sua rete locale. L'appartenenza ad un gruppo è gestita da un protocollo apposito; far

parte di un gruppo oppure abbandonarlo è possibile in qualsiasi momento. Adattarsi alla dinamicità permette di continuare ad operare anche a fronte di interruzioni di rete. Nella filosofia di IP basta comunicare il desiderio d'adesione al gruppo al router più vicino e da quel momento in poi i pacchetti destinati a quello specifico gruppo multicast verranno distribuiti anche sulla rete locale del router richiedente (e non importa se la rete locale è multicast, broadcast, unicast o qualsiasi altra cosa). Il router diventa l'unico responsabile per il gruppo multicast all'interno della sua sottorete.
La contrattazione tra host e router all'interno della rete locale avviene facendo uso di un protocollo apposito chiamato **Internet Group Management Protocol (IGMP)**, utilizzato prettamente per il multicast. Verrà in futuro inglobato in ICMP (la stessa suite di protocolli utilizzati anche in Ping e Traceroute) con il passaggio ad Ipv6. Con IGMP è il router a prendere l'iniziativa (un aspetto positivo del far lavorare unicamente il router è che tutto il lavoro viene forzato sulla rete all'interno del nodo e non viene interessato nessuno dei dispositivi di fruizione finali i quali rimangono *stupidamente* fuori dal gioco ed il cui compito è unicamente quello di visualizzare il contenuto. Così la complessità rimane al limitare della rete locale, com'era nell'interesse di DLNA). Come funziona questo protocollo: il router costruisce una lista dei gruppi multicast i cui pacchetti giungono fino a lui (è la lista di gruppi ai quali il router è iscritto, ovviamente un solo router può essere iscritto a più gruppi contemporaneamente); periodicamente invia sulla rete locale dei messaggi broadcast del tipo "*IGMP Host-membership Query"* con informazioni relative ai gruppi disponibili (se non conosce queste informazioni deve attendere che dalla sorgente arrivi un segnale di aggiornamento della lista); se nella rete locale ci sono degli host che intendono iscriversi ad un gruppo tra quelli disponibili aspettano un tempo casuale e poi mandano un messaggio di risposta "*IGMP Host-membership Report*" al router. Queste sono in sunto le regole standard: tutto dipende dalla dimensione della rete locale, se è grande e non permette il broadcast il router deve appellarsi agli host uno alla volta in maniera unicast.
Per la distribuzione dei dati si definisce una classe di indirizzi virtuali IP detta **classe D**. Sono tutti indirizzi IP che incominciano con 4 bit del tipo 1110 così definendo un ampia gamma di indirizzi utilizzabili. Si tratta di indirizzi usa e getta ed ad ogni indirizzo in classe D corrisponde un *multicast group* (i singoli indirizzi vengono allocati dinamicamente ed hanno un valore solo locale anche per il fatto che magari in una qualsiasi altra parte del mondo l'indirizzo allocato potrebbe esser già stato allocato e riservato ad altro uso). Sarà compito di chi riceverà i dati effettuare l'iscrizione al gruppo.
La condivisione delle risorse serve ad ottimizzare l'uso della banda all'interno di un gruppo. Nel caso di una trasmissione bidirezionale come quella di una videoconferenza (ma non nel caso di Skype) dove tutti sono sorgente e destinazione di dati anche se con vincoli a priori, siccome comunque si parla uno alla volta (e sorgente e destinazione sono univoci di volta in volta), si fa in modo di fare un admission su ogni comunicazione così che la banda venga sfruttata da un messaggio alla volta (ed a turni).
Il problema del realtime sta nel fatto che si deve riuscire a dare a tutti quanti la stessa possibilità di inviare e ricevere dati e che quando nella comunicazione collabora un nodo con capacità comunicative inferiori a quelle degli altri tende a degradare la qualità dell'intera comunicazione. Occorre quindi garantire le stesse possibilità trasmissive a tutti i partecipanti senza però rallentare per colpa di un solo componente la connettività, che poi diventa scadente: si

sfruttano allora delle specifiche strategie di admission control che possono essere utilizzare anche per selezionare i partecipanti alla comunicazione (si possono decidere delle soglie sotto le quali ad un nodo non è erogato alcun servizio oppure si sfrutta il *cry baby problem* per cui il nodo svantaggiato riceve tutta l'attenzione del protocollo proprio perchè svantaggiato).
Nel **routing multicast** il problema è quello di costruire un sistema di distribuzione preferibilmente ad albero per la consegna dei datagrammi tale per cui non ci siano risorse sprecate. Per far ciò si limitano al minimo i nodi non partecipanti da coinvolgere, non devono esistere cicli di nessun tipo le distanze devono comunque essere minime (problema dell'algoritmo di routing sopra il quale il multicast si appoggia in quel determinato nodo, sarà lui a scegliere la strada più corta ed efficiente a seconda delle sue specifiche impostazioni). I datagrammi verranno moltiplicati solo all'incorrere di una biforcazione: si parla di **alberi basati sull'origine** se viene creato un albero per ciascuna origine presente nel gruppo di multicast (perchè possono coesistere diverse origini) e la radice coinciderà con la sorgente; si parla di **albero condiviso** dal gruppo se viene costruito un unico albero d'instradamento condiviso per i datagrammi multicast originati da tutti i partecipanti, basato sulle distanze dove la radice è il nodo che minimizza tutte le distanze dell'albero (i dati allora vengono prima mandati al nodo radice e poi da lì vengono instradati per tutta la rete); la radice può spostarsi a seconda delle necessità e spesso viene individuata attraverso delle euristiche dedicate allo scopo.
Per costruire un albero esistono diversi approcci possibili: per un albero basato sull'origine si parla di Shortest Path Tree oppure di Reverse Path Forwarding; per un albero condiviso dal gruppo di parla invece di Minimal Spanning Tree (Steiner Tree) oppure di albero basato sul nodo centrale (in qualsiasi caso comunque sono approcci che convergono troppo lentamente e per questo godono di applicazioni solo accademiche e non pratiche e preferenziali, così si tende a scegliere l'approccio del nodo centrale a priori). Per costruire un albero ci si basa su diversi fattori come la topologia della rete che può coinvolgere tecnologie differenti e conformazioni discordanti (a stella, circolare, ecc) con soluzioni molto specifiche; la distribuzione geografica degli utenti che varia anche a seconda della quantità e delle esigenze degli utenti; la densità locale sul territorio per assecondare la quale serve un intervento umano che lavori sugli apparati attraverso dell'ingegneria pro-attiva di traffico basata comunque sull'esperienza personale. La tecnologia di rete e la distribuzione degli utenti dovrebbero influenzare pesantemente le strategie adottate.

Shortest Path Tree. Si crea un albero aggregando tutti i percorsi più brevi dalla sorgente a ciascuno dei destinatari. In pratica si applica l'algoritmo di Dijkstra nudo e crudo. È fondamentale la collaborazione da parte di router e switch che non fanno direttamente parte del gruppo multicast. L'albero a modo suo costituisce anche una variazione minima del già noto *Reverse Path Broadcast* che in questa veste prende il nome di **Reverse Path Forwarding**.
L'RPF è un parente stretto dell'RPB e sfrutta le informazioni di routing più presenti su ogni apparecchiatura locale. L'algoritmo implementato da ogni router è molto semplice: se un pacchetto viene ricevuto su una certa interfaccia che, secondo le normali tabelle di routing, viene utilizzata per mandare dati a chi spedisce, allora il pacchetto verrà preso in consegna e ridistribuito sulle altre interfacce, altrimenti verrà scartato. È ancora più semplice da implementare. Bisogna notare che il forwarding qui è fatto solo sulla rete formata da tutti gli

iscritti al servizio multicast e non su TUTTI i nodi della rete.

Alberi condivisi. In questo caso si ha un solo ed unico albero per tutti quanti. L'idea di condividere l'albero di distribuzione permette alla lunga di risparmiare globalmente delle risorse. Si può sfruttare un nodo centrale oppure si trova un compromesso per radicare l'albero in un unica sorgente, così per comunicare si manderanno messaggi sia a "monte" che a "valle" (ma così si complicano parecchio le cose). In questa maniera si ha una sola configurazione (quindi una sola tabella di routing) per i router anziché N, una per ogni possibile sorgente (ed una sola allocazione di banda se si trasmette a turno anziché N che potrebbe eccedere la capacità della rete).
Purtroppo non è possibile ottenere un albero ottimo per tutti i nodi (anche perchè non è molto interessane farlo).

Steiner Tree. È un albero a costo minimo che copre tutti i router responsabili di reti al cui interno ci sono nodi che partecipano ad un dato gruppo di multicast. Esistono eccellenti euristiche per la sua costruzione, sostanzialmente perchè nessuno lo vuole mai sfruttare (per una questione di complessità computazionale troppo alta, per la necessità di avere informazioni sull'intera rete e perchè dev'essere ricalcolato ogni volta che c'è una variazione del gruppo, oltre per i suoi tempi di convergenza imbarazzanti).

Alberi con nodo centrale. Un router viene identificato quindi eletto come il "centro" della rete e lo si usa per la costruzione dell'albero in maniera simile allo *spanning tree*. Questo router centrale va scelto con attenzione e furbizia quindi lo si fa manualmente (e non via software in maniera automatizzata). L'aggancio di qualsiasi altro router a quello centrale avviene nella stessa maniera di come accade con l'albero di copertura: un router che vuole far parte di un certo gruppo manda un messaggio di *join* verso il router centrale ed il punto d'aggancio sarà il router centrale stesso o il primo ramo già facente parte del gruppo che il messaggio incontra. Non c'è bisogno di essere iscritti al gruppo multicast per ricevere e mandare un pacchetto multicast.

Tecniche di distribuzione dei dati

Dopo la costruzione di un albero esistono diverse tecniche grazie alle quali inviare dati al suo interno che dovrebbero tenere conto della densità distributiva degli utenti (ma non sempre succede).

1. **data replication**: sulle biforcazioni e sull'ultimo router foglia (quello che poi s'interfaccia con la sua rete locale) si replicano i dati e si manda un pacchetto separato ma uguale per tutti a ciascun destinatario;
2. **flooding**: si mandano pacchetti broadcast per servire tutti gli utenti in un solo colpo, tipico ed auspicabile solo quando si hanno reti piccole e diverse risorse da sprecare (ma capita che non si faccia solo in una piccola rete locale, anche in una rete più ampia come quella di un palazzo).

In entrambi casi la cosa più importante è contare preventivamente gli iscritti ai gruppi di multicast e se questi sono molti si preferisce il flooding altrimenti si opta per il data replication (cosa che oggi purtroppo non si può ancora fare).
Nel mondo delle possibilità reali si utilizzano protocolli per creare alberi e decidere su di essi quale distribuzione dei dati scegliere di sfruttare. Solitamente

il protocollo si occupa di scegliere a priori un certo tipo di albero e quindi, di conseguenza, un certo tipo di distribuzione di dati (e non varia, ciascun protocollo si occuperà sempre e solo di un determinato tipo di albero senza prendere in considerazione altre alternative a seconda della topologia e delle possibilità della rete).
Il protocollo **Distance Vector Mutlicasting Routing Protocol (DVMRP)** implementa un albero basato sull'origine (che coincide con la sorgente dei dati) facendo uso di RPF per distribuire i dati. Come suggerisce il nome, questo protocollo si basa (e si appoggia nei fatti) sull'algoritmo di routing *distance vector* per calcolare il percorso minimo verso la sorgente. Con l'ausilio di appositi algoritmi di potatura è possibile tagliare i rami secchi ottimizzando e riducendo l'albero adattandosi così alla dinamicità dei gruppi (accade spesso che un host continui a rimanere collegato anche se l'erogazione del contenuto multimediale è stata troncata da tempo, così si mandano continui messaggi d'accertamento grazie ai quali è possibile capire se quel dato ramo è potabile o meno), in questa maniera ogni router mantiene una lista di tutti i next-hop "a valle". È un protocollo che oramai si trova in quasi tutti i router commerciali (di una certa fascia) ed è standardizzato dall'RFC 1075.

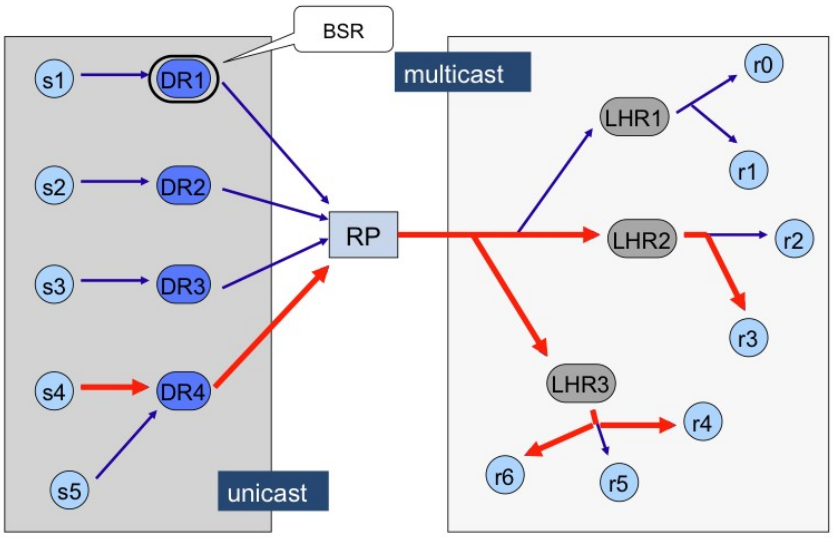
Richiami sugli algoritmi di routing. La rete è formata da una quantità di diversi *autonomous system* (AS), così esistono due tipologie di algoritmi di routing che hanno a che fare con loro: gli algoritmi per la comunicazione interna tra i nodi facenti parte di uno stesso AS sono detti Interior Gateway Protocol (IGP) mentre gli Exterior Gateway Protocol (EGP) si occupano di effettuare la corretta comunicazione tra AS differenti. I protocolli di tipo EGP devono tenere conto delle differenze tra i diversi AS che possono essere non solo tecnologiche ma anche politiche (contratti di licenza, appartenenza, ecc). Poiché gli AS non sono molti e non se ne creano né se ne distruggono in maniera frequente, di EGP ne esiste uno solo che si chiama Border Gateway Protocol (BGP) e che utilizza delle tabelle di instradamento fisse. DVMRP è un protocollo IGP. Per fare multicast con utenti che risiedono fuori dall'AS corrente si devono applicare delle estensioni per renderlo gerarchico (così da effettuare prima il multicast all'interno dell'AS, poi al suo esterno quindi all'interno di quello di destinazione) comunque sia, DVMRP compie un multicast incompleto e maldestro anche se lo si sfrutta ugualmente: tenta di adattarsi alla dinamicità dei gruppi (perchè gli utenti tendono ad iscriversi ai gruppi ma non a disiscriversi, così si utilizzano dei sistemi di *keep alive* per controllare ed eventualmente forzare la cosa: non disiscriversi significa bruciare delle risorse utili altrimenti impiegabili).

Un'altro dei protocolli che si utilizzano è **Protocol Indipendent Multicast (PIM)** il quale costituisce un protocollo di routing multicast in grado di funzionare utilizzando un qualunque protocollo di routing unicast come supporto (si chiama *Indipendent* appunto perchè è indipendente dall'algoritmo di instradamento sui router sottostanti). È un protocollo diffusissimo e molto sfruttato, può essere utilizzato per il routing sia all'interno che all'esterno di una sottorete (inter-domain ed extra-domain) senza che sotto ci sia per forza una rete eterogenea a livello di protocolli di routing; se utilizzato come extra-domain (EGP) allora si riesce ad interoperare con qualsiasi altro protocollo di tipo inter-domain (IGP). Gli AS poiché sono indipendenti tra loro non hanno interesse a

comunicare a livello amministrativo (gli amministratori di un AS non parlano mai con quelli dell'AS vicino) ed una volta non si trattava di un problema molto sentito perchè il multicast era fatto solo all'interno degli AS. Con PIM ci sono una serie di euristiche per la costruzione di alberi ad alto livello, tuttavia le foglie esulano dal controllo di PIM e possono essere complicate a piacere (perchè a PIM non interessa come i dati arrivano a destinazione e compiono l'ultimo passo del loro viaggio, lui si occupa solo di farli arrivare fino alle foglie dell'albero). Gli alberi di PIM sono tutti monodirezionali e si distinguono i nodi di sola ricezione ed i nodi di solo invio. Sulla carta si parla di due distinte modalità di funzionamento: *Sparse Mode (PIM-SM)* e *Dense Mode (PIM-DM)* a seconda della distribuzione degli host finali, tuttavia nella realtà si utilizza solo PIM-SM perchè è l'unico che esiste davvero:

1. PIM-DM: è stato progettato ed ottimizzato per reti densamente popolate da utenti di un certo gruppo, viene fatto costante uso di flooding;
2. PIM-SM: progettato per essere applicato su reti scarsamente popolate, vede l'utilizzo massivo di data replication.

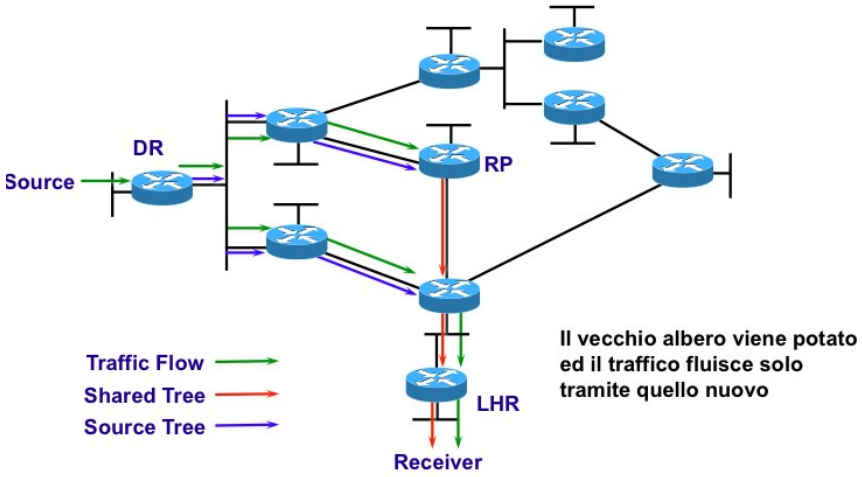
La modalità di funzionamento è impostata a priori e non esistono, per ora, metodi per passare dinamicamente dall'una all'altra.
Vederemo come si comporta il PIM-SM.
PIM divide tra loro nodi sorgenti e nodi destinatari; tutte le sorgenti si trovano in un segmento di rete dove il multicast non è supportato (quindi comunicano solo via unicast) mentre le destinazioni sono tenute a comunicare tra loro mediante multicast (per una questione di opportunità). Tra i due blocchi di rete è necessario individuare un **randesvouz point router (RP)** che lavora come coordinatore della trasmissione (in teoria ce ne dovrebbe essere uno per ogni gruppo di nodi sorgente e se si dovesse guastare sarebbero guai anche se non è del tutto vero). L'RP è il punto d'ingresso all'albero di distribuzione, sia per iscriversi ad un gruppo che per inviare dati, è relativo ad uno specifico gruppo multicast ed esiste solo in SIP-SM; la sua funzione è quella di ridistribuire i dati tramite un albero monodirezionale. Ogni sorgente deve poi possedere un router di riferimento detto **designed router (DR)** che rappresenta l'unico router direttamente connesso alla sorgente che invia i dati; generalmente è comune a tutta la sottorete e può corrispondere ad una o più sorgenti. Le reti d'accesso sono tra loro disgiunte. Il DR non sa nulla sui dati che viaggiano all'interno della rete e quando li prende in consegna semplicemente li sparge nella rete. Si ricorda che i protocolli all'interno degli AS sono diversi a seconda dell'AS in cui ci si trova. Un equivalente del DR è posizionato anche al termine dell'albero ma non lo gestisce direttamente PIM (è da lì in avanti che il protocollo “se ne lava le mani” e delega tutto alla gestione dei sistemi terminali), tale router è detto **last hop router (LHR)**: è il router che è direttamente connesso con l'host che riceve i dati e si occupa da solo della distribuzione multicast sulla sottorete di destinazione; in generale per reti piccole il DR coincide con il LHR ma questo non è sempre vero. Il DR comunica con uno o più di questi router, mentre il **boot strap router (BSR)** è una specie di DR che si occupa della creazione di tutti gli altri DR e dei RP, in maniera semplicistica inizializza il sistema ed assegna i compiti a ciascun DR dopodichè esso stesso funge da DR normale, in più controlla che tutto fili liscio e se avviene qualche guasto lo segnala e fa in modo di risolverlo assegnando alla sorgente un nuovo DR (per quello quando si diceva che se si guasta un DR sono guai non è del tutto vero).
È dal RP che parte la comunicazione multicast (prima è unicast) dal cui router si dirama un albero verso ogni LHR a partire dai quali nascono diversi sottoalberi.



La creazione dell'albero di PIM avviene in due momenti separati. Si incomincia a lato ricevente dove un host notifica al suo LHR che intende partecipare ad un certo gruppo; l'LHR allora rintraccia il RP opportuno per quel dato gruppo e manda un messaggio di *join* al RP . Il percorso effettuato dal messaggio di join attraverso i router intermedi determina il ramo dell'albero di distribuzione dal RP alla sottorete (della quale LHR è router rappresentativo) di cui fa parte l'host che si vuole iscrivere. A lato sorgente un host/sorgente notifica al suo DR che intende partecipare ad un certo gruppo così il DR rintraccia il RP opportuno per quel gruppo (perchè di RP ce ne possono essere più di uno a seconda del gruppo e dei casi), allora il DR forwarda il messaggio di *register* all'host RP (che riceve il messaggio e valuta se esiste già un albero di distribuzione per il gruppo a cui fa riferimento l'host che vuole trasmettere: se il gruppo esiste già semplicemente aggiunge una nuova sorgente alla lista di quelle già presenti). Infine a lato destinazione viene creato un albero avente come radice l'RP e come foglie tutti gli host in ascolto, viene quindi distribuito su tutto l'albero il pacchetto *request* ed i router per cui il pacchetto transita creano all'interno delle loro tabelle delle entry di collegamento diretto tra LHR del ricevente e DR della sorgente. Alla creazione di tali entry i riceventi mandano un certo messaggio di *join* verso la sorgente (*switchover*) ed il risultato finale è la trasformazione dell'albero avente come radice l'RP in un albero a cammino minimo tra il DR e tutti gli LHR.
Per riassumere: si sono creati due alberi uno tra RP e sorgenti e l'altro tra RP e LHR. Ad un certo punto è possibile creare un unico grande albero unendo questi due ed ottimizzandoli: dopo che una nuova sorgente si è palesata questa manda un messaggio unicast al DR che provvede ad iscriversi al RP, intanto l'RP si registra in multicast presso il DR e si viene a creare l'albero tra i due che fa fluire i dati troncando definitivamente il collegamento unicast (attenzione: ma solo ed esclusivamente dopo la creazione del secondo albero... il primo albero è quello tra la sorgente e RP ed è stabile e difficilmente cambierà in futuro, mentre il secondo è quello tra RP e destinazione che però è molto fluido e cambia di continuo ma non è controllato da PIM).

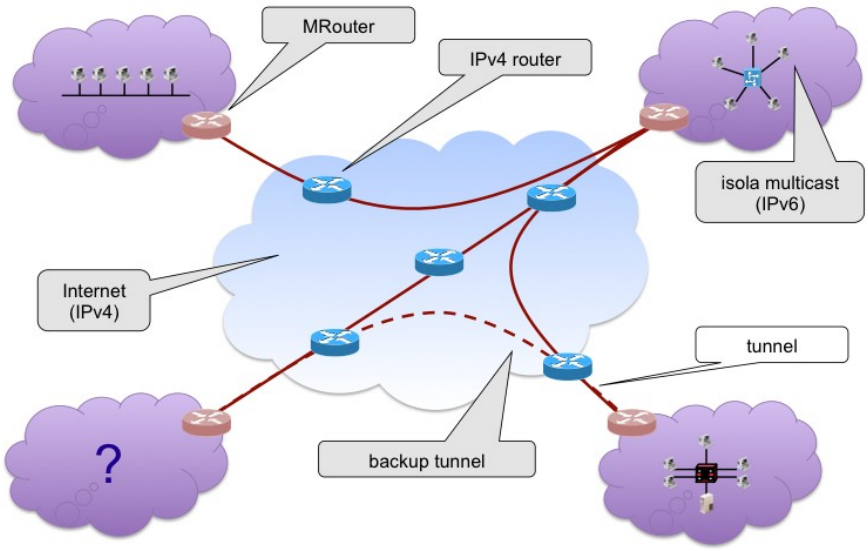
Switchover. L'RP comunica a LHR che esiste una certa sorgente multicast e che sarebbe opportuno che LHR si registrasse presso di lei (in questa maniera verrà a formarsi un terzo albero tra LHR e sorgente) così i dati passano anche per questo nuovo collegamento. Si ha così una sorgente, una destinazione e due

alberi che le mettono in comunicazione, così si hanno dei router che ricevono due volte gli stessi identici dati, allora sarà compito suo scremare tutte le informazioni e scegliere quali fare passare per non sprecare troppe risorse. Si tratta comunque di una situazione transitoria, perchè LHR si disiscrive subito dall'RP conservando il nuovo canale di comunicazione: RP riceve allora dei dati che non può più tramandare e chiede la disiscrizione al DR.



La gestione di alberi come questi è molto complicata per via dei molti passaggi che devono funzionare correttamente anche in caso di cambiamenti di topologia della rete. L'albero di distribuzione di un certo gruppo deve rimanere stabile anche in caso di crash e reboot di apparati, crash e reboot di nodi, timeout dovuti alla congestione della rete. Periodicamente vengono inviati dei messaggi di *join* e *prune* per evitare il timeout sulle tabelle ed allo scadere del timeout un ramo dell'albero che non si è mai fatto sentire viene eliminato dalle tabelle di distribuzione.
La distribuzione dei dati è anch'essa difficoltosa: l'RP fa sempre parte del gruppo multicast ma non è detto che sia interessato a ricevere i dati del gruppo, utilizza tabelle per lo smistamento contenenti informazione rognose da gestire e reperire che comunque non distinguono unicast da multicast. Quando un pacchetto contenente informazioni arriva ad un router DR questo cerca all'interno delle sue tabelle un riferimento alla sorgente ed al gruppo, oppure al gruppo ed ad una qualsiasi sorgente oppure all'RP corrispondente all'indirizzo di destinazione e qualsiasi sorgente e gruppo. In caso di fallimento della ricerca il pacchetto viene semplicemente scartato. Altrimenti in caso di esito positivo, se l'interfaccia di rete dalla quale ci si aspetta i dati corrisponde con quella da cui effettivamente il pacchetto è arrivato, questo viene instradato sull'interfaccia di uscita presente in tabella, altrimenti viene scartato.
Le cose tendono a complicarsi molto se un pacchetto viene instradato mentre si compie la transizione di stato dell'albero: se si perdono pacchetti mentre avviene lo switchover li si deve rigenerare ma contemporaneamente si deve far tornare l'albero allo stato precedente (quindi far arretrare di un passo l'intera infrastruttura).
Esistono nodi che possono mandare dati contemporaneamente e se questo accade vanno presi seri provvedimenti, come l'elezione esplicita del DR responsabile per una certa sottorete (capita quando si ha un DR per diverse sorgenti) o l'individuazione di un sistema per la determinazione di cammini univoci quando più di un percorso è disponibile o la soppressione di messaggi di

join e *prune* nel caso vi siano messaggi di controllo duplicati.
Si solleva però un problema (valido per ogni sistema multicast e non solo per PIM): se lungo il percorso ci sono degli apparati che non sono in grado di gestire indirizzi di classe D o che non implementano IGMP? I pacchetti finiscono sistematicamente scartati in maniera totalmente silente (quindi non si può sapere se questo accade o meno, e si finisce a inviare pacchetti all'infinito senza combinare nulla). La soluzione è offerta dagli **MBone** (*Multicast Backbone of the Internet*) che costituiscono una rete virtuale progettata per risolvere il problema di fare multicast su internet in un momento in cui non tutti i router installati erano in grado di gestire indirizzi di classe D, così si bypassano le zone non multicast (progettato in un momento in cui la maggioranza di internet non era assolutamente multicast). Vengono costruite delle isole (come succedeva per IPv6 e diff-serv) periferiche dove il multicast viene supportato con qualsiasi protocollo noto; la comunicazione tra le varie isole avviene attraverso dei collegamenti detti tunnel nei quali viene fatto circolare il multicast IP incapsulato in IP classico. L'accoppiamento più sfruttato è quello tra multicast IP e DVMRP per isole piccole e PIM-SM per isole medio-grandi.

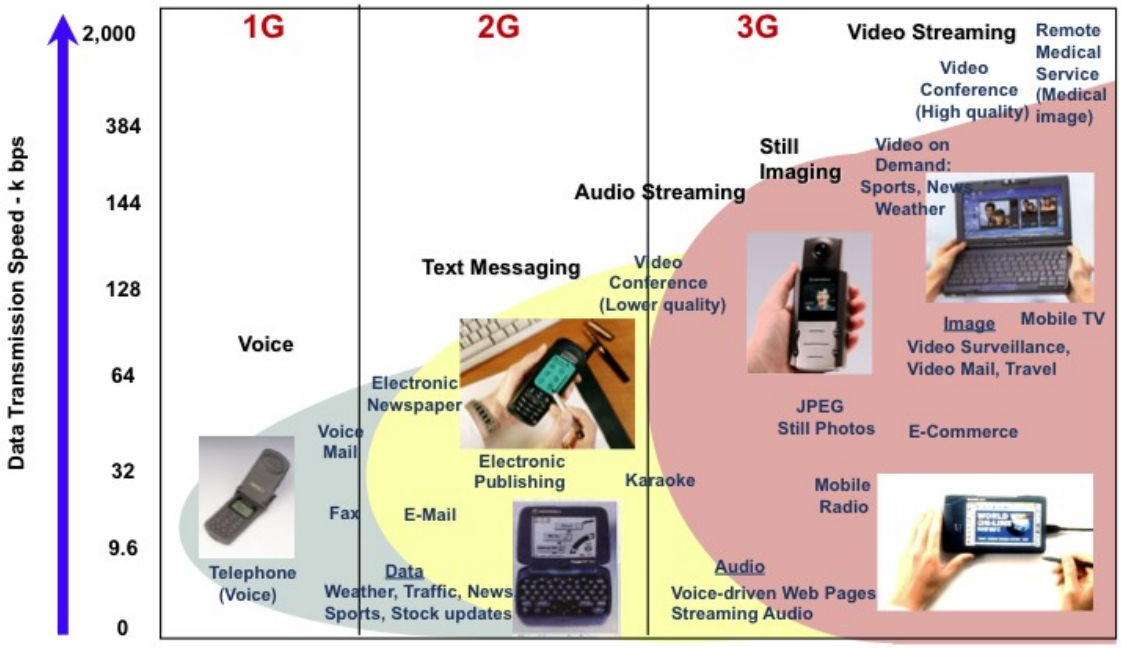


Gli Mrouter sono i punti terminali dei tunnel (scritti tutti a mano su dei file di configurazione) attraverso i quali è possibile accedere alle isole; si occupano di incapsulare e de-capsulare i pacchetti IP multicast da IP unicast e di distribuirli sulla rete locale (e viceversa). Si possono realizzare con un host su cui gira un software chiamato *mrouted* oppure con un apparato di rete in grado di gestire i tunnel di MBone.
La topologia di MBone si compone di un misto tra architetture a stella ed ad albero e si tratta di reti regionali collegate tra una ragnatela molto fitta e caotica di tunnel (perchè si voleva che queste isole fossero fortemente connesse). Internamente una rete regionale è organizzata ad albero ed all'interno delle reti regionali risiedono le isole delle singole organizzazioni disposte secondo una topologia a stella.
Quella di MBone è stata una diffusione a macchia di leopardo che, con il passare del tempo, si è espansa tanto da ridurre al minimo (fino alla scomparsa) le zone dove il multicast non era supportato. IPv6 sta sfruttando la stessa tecnica per subentrare ad IPv4.

Capitolo 11
Wireless Multimedia

Wireless Multimedia

La **wireless multimedia** è una contraddizione di termini. Si può vedere il wireless come la tecnologia del presente (e del futuro): permette il dialogo tra service provider differenti al livello della telefonia mobile. Il multimedia è molto più complesso poiché costituito da diversi flussi multipli di dati.
La storia dello sviluppo della telefonia cellulare si può riassumere in generazioni tecnologiche, ciascuna con le proprie caratteristiche.

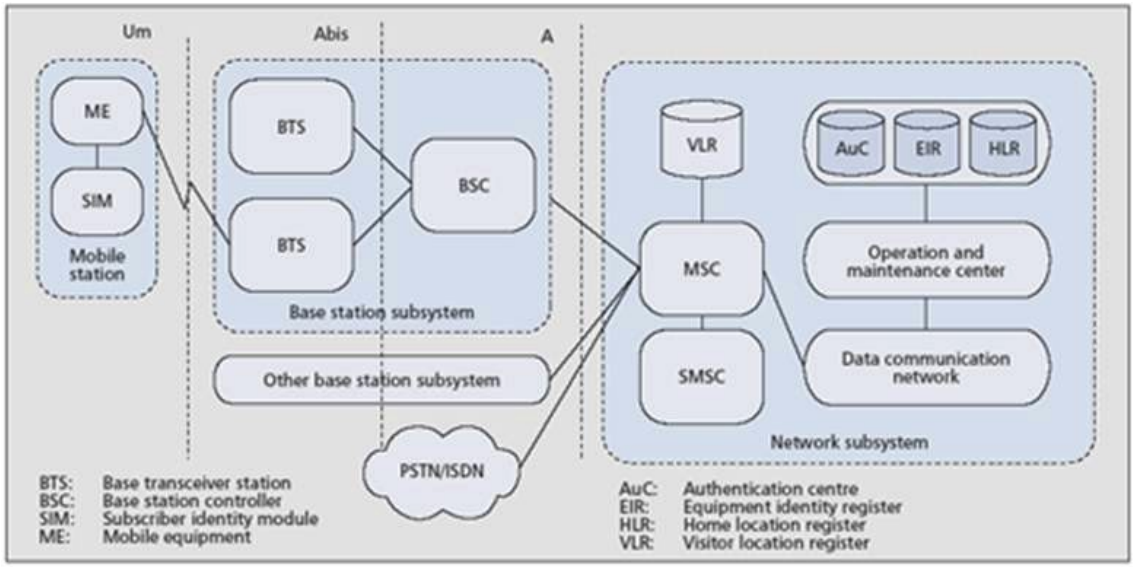


1. **Prima generazione (1G).** Si incomincia a trasferire la telefonia un tempo solo fissa su supporti indipendenti dalla cablatura (wireless) rendendola quindi mobile. In Italia comporta la comparsa di TAX, tecnologia ora completamente in disuso che persiste con ancora 40 abbonati a causa dei quali Telecom deve continuare a fornire il servizio (si tratta di utenti che basano la loro attività commerciale su TAX e non sono per nulla intenzionati a dismetterla per passare a Telecom); la presenza di TAX rende difficile gestire le telefonate da parte del gestore vigente. Si passa poi all'invio di email ed al traffico dati attraverso il modem collegato al telefono cellulare e non più direttamente alla rete telefonica cablata;
2. **Seconda generazione (2G).** In Italia compare il **GSM** che mirava a gestire e cambiare le risorse utilizzate dalle dorsali telefoniche e non direttamente i dispositivi mobili;
3. **Terza generazione (3G).** Si parla di servizi di trasmissione voce mescolati ai servizi dati.

Nella seconda generazione si commerciava in servizi di voce dove i servizi dati costituivano un valore aggiunto: durante la terza generazione sino ad oggi è la

voce a costituire un valore aggiunto ai dati. Sempre e comunque è il servizio dato dall'operatore telefonico a subire dei mutamenti e delle migliorie: durante la prima generazione non si parlava neppure di servizi a commutazione di pacchetto emersi poi durante la seconda generazione assieme ai protocolli IP → la vera innovazione è sempre stata direttamente sulle dorsali degli operatori telefonici e mai sui dispositivi che ci hanno guadagnato, di riflesso, solo un certo aumento di banda a disposizione, perchè fondamentalmente la connettività ce l'avevano anche prima.

Rete GSM. Si tratta di un'infrastruttura tecnologica pensata per consentire ad un dispositivo wireless di compiere una telefonata.



Il problema del wireless è la gestione dell'ultimo pezzetto della rete, che si può identificare con la dicitura “ultimo miglio”: l'ultimo miglio della comunicazione cellulare è costituito dai **Base Transciever Station (BTS)** che fungono anche da *access point wireless* (per intenderci, sono quelle stecche bianche che spesso si vedono sui tetti dei condomini). Un gruppo di questi BTS può estendersi in un area di 300/500 m di diametro e fanno capo ad un **Base Station Controller (BSC)** il quale gestisce le telefonate di tutto il quartiere. Nei BSC converge tutto il traffico mobile dei BST che finisce poi nella dorsale telefonica attraverso un **Mobile Switching Center (MSC)** il cui compito fondamentalmente è quello di instradare tutte le chiamate. Il **Network Switching Subsystem (NSS)** è composto da tre differenti database ai quali l'MSC si appella per conoscere i dati degli utenti che effettuano le telefonate che sta instradando: **AUC** è il database che riporta i nomi degli abbonati all'operatore telefonico in relazione al numero identificativo della loro carta SIM (e non al loro *numero di telefono*, che non fa altro se non costituire una specie di GUI interattiva che ci aiuta a ricordare meglio il numero associato ad ogni utente) quindi, in sostanza, l'elenco di utenti che sono autorizzati a registrarsi sulla rete; **HLR** contiene entry per ciascun iscritto al servizio di telefonia mobile con le corrispondenze tra SIM e la posizione geografica effettiva a ridosso del BTS più vicino, e **VLR** invece è dedicato agli operatori telefonici stranieri ed ai loro iscritti che si trovano ad utilizzare la rete di un operatore telefonico che non è specificatamente il loro. L'MSC può allora collegarsi alla rete fisica.

Se si ha intenzione di fare traffico dati verso un qualsiasi server a partire da questa configurazione si devono tenere in conto altri fattori ed altri dispositivi tecnologici, come il **GMSC Gateway** per la gestione dei dati (perchè all'epoca collegare direttamente ad internet l'operatore telefonico era considerato al pari di un sacrilegio). Le soluzioni erano: utilizzare un telefonino con un modem integrato oppure si poteva fare traffico dati tramite GMSC così da chiedere in prestito un modem alla rete dell'operatore telefonico. Comunque fosse, dall'altra parte del PSTN doveva esserci per forza un altro modem, non importa di chi fosse. A partire dalla seconda generazione i modem si collegano direttamente all'MSC.
GPRS → si colloca tra la seconda e la terza generazione. Della precedente GSM la parte relativa alla segnalazione non va più tanto bene e bisogna ri-progettarla, poiché gestiva più canali contemporaneamente di upload e download (alle volte totalmente inutili). Gli **SGSN Gateway** si collegavano alla struttura GSM e facevano da ponte verso gli altri operatori oppure verso la rete a protocollo IP per eliminare la necessità della presenza forzata di tutti quei modem. Il tutto col passare del tempo è stato spinto all'eccesso tanto che oggi per il traffico voce si utilizza massivamente VoIP unitamente a SIP per instaurare le chiamate mentre per il traffico dati e tutti gli altri servizi si utilizza esclusivamente la rete IP (ES: gli SMS sono solitamente inviati verso un dispositivo SMSC che in sostanza è un computer con in uscita una scheda di rete: all'arrivo di un SMS viene creata una socket di collegamento con il server che poi instrada il messaggio).

Nella **quarta generazione (4G)** la voce ha sempre meno peso a discapito del traffico dati con una banda sempre maggiore per effettuare telefonate con una certa quantità di valore aggiunto di natura prettamente telefonica. Tuttavia non serve tutta questa banda: in futuro le reti wireless e le reti broadband wireless convergeranno (forse) su una tecnologia unica come quella del 4G tuttavia esistono diverse problematiche da risolvere; le reti cablate convergeranno allora sulla fibra ottica, tuttavia si tratta di una soluzione che sposerà le esigenze delle grandi aziende e non quelle casalinghe e domestiche (dove l'assenza dei cavi è la scelta prediletta).

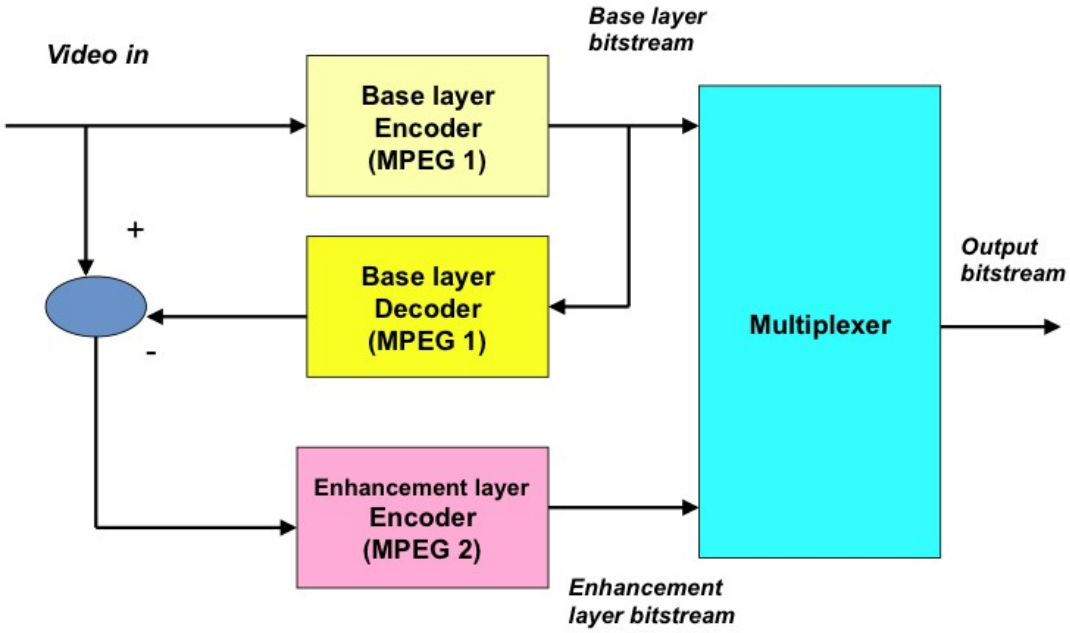
QoS per il wireless

La QoS per le reti wireless è difficile da definire. Intanto è sbagliato pensare di aver a che fare con delle reti tradizionali, quelle wireless presentano un comportamento dei bit e dei pacchetti inviati molto diverso. Le reti wireless tendono a perdere dati perchè i pacchetti sono e devono essere trattati in maniera diversa rispetto le reti cablate; ci sono, per esempio, molte interferenze e la contrattazione degli indirizzi MAC tende a portare via moltissimo tempo alla comunicazione, così vale la pena di comunicare solo su canali a frequenze diverse. La banda nel wireless (come in tante altre situazioni) non può essere semplicemente espansa ed aumentata all'occorrenza, bisogna attendere il ricambio generazionale poiché si tratta di un problema prettamente tecnologico. Il dispendio energetico, in ultimo, è un fattore che va sempre preso più in considerazione specialmente per quanto riguarda la comunicazione cellulare e che incide moltissimo sull'utenza finale del servizio (cose che sul cavo non avevano tutta questa rilevanza).
In sintesi bisogna ripercorrere le tappe già viste per la rete tradizionale cercando di cambiare la prospettiva con la quale le si guarda: cos'è cambiato rispetto a

quanto visto per le reti cablate? Il canale è più difficile da trattare: ci sono problemi di allocazione delle risorse/frequenze e di codifica delle informazioni per tentare di minimizzare gli errori di trasmissione/comunicazione. La *TDMA* aiuta in questa direzione costituendo un sistema di collegamento a livello di data link privo di collisioni (ed anche di contrattazione per le risorse, cosa che velocizza di molto l'instaurazione del collegamento). Nella codifica si parla di codifica di linea (già affrontata), di codifica di sorgente (che cerca di disperdere meno informazione possibile creando simboli grazie ai quali fare una correzione degli eventuali errori), codifica di canale (grazie alla quale trovare combinazioni di simboli per riuscire a sfruttare la massimo la capacità del canale). Si deve trovare il miglior compromesso per la convivenza della codifica del canale trasmissivo con quella della sorgente dei dati: se l'obiettivo si sposta sulla codifica della sorgente basta ottenere la distorsione minore possibile dato un certo bit rate, se invece l'obiettivo è ottenere la codifica del canale basta conservare le informazioni in maniera più affidabile ad una frequenza il più possibile vicina alla capacità trasmissiva del canale.
Il **principio di separazione di Shannon** sostiene che è possibile considerare in maniera indipendente le codifiche di canale e sorgente (costruendo una codifica a sé stante) senza nessuna perdita di prestazioni; per cui è possibile tenerli separati e cercare di ottimizzare i loro parametri congiuntamente (quindi bilanciare l'overhead di canale e sorgente). Il principio di separazione vale solo per le comunicazioni punto-a-punto.
Tuttavia esistono difficoltà anche a livello di codifica: così si creano dei sistemi ad hoc per la trasmissione di dati audio e video. Lo **Scalable Video Coding** è un approccio che permette ad un sistema di streaming di essere utilizzato su reti d'accesso con diverse tecnologie (e diverse caratteristiche) ed in condizioni di variazioni significative di banda disponibile. Tale approccio fa sì che un video si adatti in tempo reale alle variazioni di banda disponibile ed ai dispositivi pur facendo uso dello stesso contenuto registrato: si utilizzano dei flussi paralleli di dati multipli trasmessi in contemporanea (*Layered Video Coding*). Di questi flussi alla fine faccio passare solo quelli supportati e gestiti dalla rete a seconda delle sue condizioni correnti; sul cablato non si fa la stessa cosa perchè se nel wireless il problema fondamentale sta nell'ultimo miglio (dove si trova sempre il peggiore tra tutti i colli di bottiglia) nel cablato potrebbe essere ovunque, inoltre nel wireless ha senso avere più canali per via della presenza di diverse frequenze e nel cablato invece non ha nessun significato perchè non è possibile nessuna separazione.
Esistono diverse tecniche per rendere possibile la scalabilità:

1. **Data partitioning.** Tecnica che viene utilizzata quando è disponibile più di un canale di trasmissione. Non si tratta di una vera e propria codifica scalabile: il mittente suddivide il flusso dati in due o più byte-stream a seconda dei canali disponibili ed il ricevente li ricompone. Il guaio è che può capitare che sia fondamentale ricevete tutti gli stream messi a disposizione e non solo alcuni di essi, invalidando così l'idea di una eventuale scalabilità verso un sistema con molti canali ma con poca banda;
2. **SRN scalability.** Si tratta di una tecnica dinamica caratterizzata da un approccio incrementale. Si suddividono i dati in un flusso di base al quale viene aggiunto uno o più flussi di supporto per arricchire i dati e renderli più dettagliati. Il numero di flussi da inviare viene stabilita sulla base del valore di SRN rilevato (l'SRN costituisce un rapporto matematico tra la

quantità di dati inviati e la quantità di dati ricevuti corrotti, sulla rete cablata questo rapporto tende all'infinito mentre nel wireless raggiunge cifre considerevoli. A seconda del suo valore l'invio di dati va accelerata oppure rallentata).



Si ha un video in input (il flusso di base) che viene codificato con MPEG1 (perchè è un formato veloce compatibile con le trasmissioni in tempo reale) il quale poi passa attraverso un multiplexer che lo colloca nella rete verso il ricevente. Con questa specifica codifica si spreme il più possibile qualsiasi informazione presente nel video: l'idea è quella di alleggerire la trasmissione al massimo perchè sia innanzitutto efficiente e solo in un secondo momento visivamente apprezzabile (dopotutto si tratta di video che andavano guardati in display davvero piccoli ed infimi). Tuttavia il video, prima di passare per il multiplexer, viene copiato e decodificato (così da ottenere una copia esatta di ciò che il ricevente sta guardando). Si fa allora una differenza frame by frame tra il video originale e quello decodificato per ricavare tutte le informazioni aggiuntive che mancano all'utente: queste vengono codificate tramite MPEG2 (che è un po' più sofisticato del fratello MPEG1) e vengono spedite al multiplexer in aggiunta al flusso principale. Il multiplexer controlla che il ricevente goda di abbastanza banda per ricevere entrambi i flussi in maniera alternata, se non è così invia solo il primo che è quello di base (che viene comunque inviato a prescindere qualsiasi cosa accada). Si tratta di un sistema che funziona abbastanza bene (e che è abbastanza simile al piggyback).

3. **Spatial/temporary scalability.** Tutte le informazioni di ridondanza spaziale o temporale vengono ridotte oppure eliminate in base alla banda disponibile (dipende infatti dalle risorse del canale). Si prende un GOP MPEG nel quale si inviano il frame I e poi i frame P e B solo se le risorse lo consentono (quindi potrebbe succedere che non venga spedita tutta la risorsa): in questa maniera si invia tutta l'informazione possibile. MPEG standard tuttavia non supporta questo comportamento quindi occorre definire delle codifiche accessorie e specifiche perchè un qualunque B-

frame possa essere soppresso senza creare artefatti (o creandone il minor numero possibile).

Per la banda trasmissiva c'è veramente poco da fare perchè dipende solamente dalla tecnologia e come già detto per aumentare la connettività serve attendere il salto generazionale (all'inizio c'era solo Aeronet → 802.11 dal quale poi sono derivati tutti gli altri standard distinguibili dalle diverse lettere dell'alfabeto). Il link wireless è il collegamento più debole del sistema trasmissivo perchè la sua capacità è limitata da diversi fattori fisici come lo spettro disponibile, il rumore di fondo, le interferenze ed il fading (spesso il segnale rimbalza e si disperde irraggiandosi dall'access point, generando bit che possono entrare in collisione con altri bit ed influenzandone traiettoria e natura).

La disponibilità di banda è una caratteristica legata all'area d'interesse e dipende dal numero di nodi che ricevono/trasmettono in quella determinata zona. Spesso viene influenzata anche dai nodi che non sono all'interno del raggio trasmissivo. Su una stessa rete si hanno nodi distinti che possiedono una visione diversa della capacità effettiva della rete e sulla sua disponibilità di banda.

Ma quando si fa una *call admission* a quale banda nello specifico si deve fare riferimento? Effettuare una call admission non è cosa da poco: se il percorso coinvolge più nodi potrebbero esserci delle interferenze tra i vari hop; inoltre i nodi si muovono e mutano, entrano ed escono dalla rete e non si sa mai con sicurezza dove si trovano. A causa delle interferenze la banda richiesta ad ogni nodo intermedio è un multiplo di quella effettivamente necessaria: il valore dipende dal livello di contesa in quella zona. Il livello di contesa è dato dal numero di nodi facenti parte dello stesso percorso che, nei confronti di un altro hop, devono contrattare l'accesso al mezzo trasmissivo. Non è detto che esista per forza un percorso che garantisca la buona riuscita della call admission.

Per quanto riguarda il consumo energetico si deve sempre fare i conti con la capacità di carica delle batterie dei nodi che, per via della loro natura mobile, spesso non hanno possibilità di allacciarsi alla rete elettrica; perchè sprecare la propria carica residua per permettere di far passare traffico di nessun interesse per un certo nodo attraverso sé medesimo e quale impatto potrebbe avere sulla codifica l'essere a secco con la batteria? Nella fase di call admission un percorso è praticabile se e solo se tutti i dispositivi intermedi hanno abbastanza energia disponibile, quindi bisogna tenere sempre presente l'autonomia energetica dei nodi che costituiscono una rete (si preferirà fare la call admission attraverso nodi che dichiarano un'autonomia simile tra loro e superiore alla quantità di tempo richiesta dalla connessione); durante la codifica dei dati la codifica stessa dev'essere il meno onerosa possibile per la CPU anche se le codifiche pensate per il wireless e per i dispositivi mobili sono più complesse di quelle tradizionali, inoltre i nodi devono consentire oltre alla codifica anche la trasmissione e la decodifica dei dati. Il consumo energetico diventa parte integrante dei parametri della qualità di servizio. Web altruistico → si ha una contrattazione delle risorse web tra l'energia che si mette a disposizione al sistema ed i dati/servizi di rete richiesti.

Le reti wireless sono sì multi-servizio ma allo stesso tempo sono anche multi-tecnologia: se ci si pensa spesso si ha solo l'ultimo hop che è veramente wireless. Tuttavia può capitare che sia wireless anche qualche altro collegamento intermedio specialmente quando non è possibile per diversi motivi fare un vero collegamento cablato (in cima ad un monumento o sulla facciata di una chiesa, dove la posizione è strategica ma è impensabile costruire un impianto a vista). Si

tratta di collegamenti wireless monodirezionali. È irrealistico pensare che una rete di distribuzione dati, in generale, utilizzi una sola tecnologia (ovvero il wireless): spesso si ha a che vedere con tratti di *backbone* che forniscono l'allacciamento al servizio ed al traffico dati completamente cablati; *WiMax* che costituisce un wifi a lunga gittata che, per la sua fruizione, mette a disposizione dei dispositivi casalinghi forniti di due antenne, una per la ricezione del servizio WiMax e l'altra wireless per la ripetizione del segnale in locale, molto costoso a livello energetico e di radiazioni; la copertura *UMTS* in Italia è discreta in città ma meno efficiente in periferia (campagna) tant'è che chi crea applicazioni mobili tende ad accontentarsi anche del servizio *GPRS* quando l'utilizzo di UMTS non è possibile.

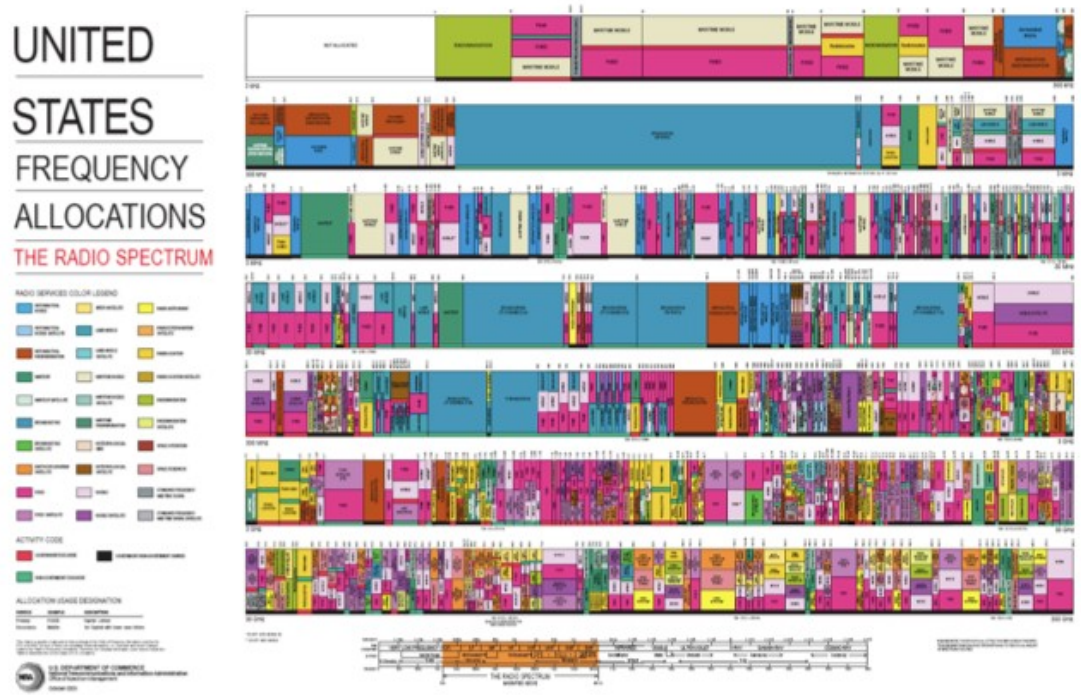
Wireless in ambito urbano

È in città che esiste la maggior richiesta di servizi di connettività. Si è pensato di sfruttare la rete già esistente all'interno degli edifici cittadini (ADSL) fisica e pre-esistente come backbone per dare connettività anche ai dispositivi che si trovano in strada (o comunque all'aria aperta). Una delle applicazioni di questa idea riguarda infatti le comunicazioni wireless con i veicoli motorizzati come le automobili, dove le aziende produttrici fanno grandissimi investimenti in ricerca → 802.11p è lo standard wireless che si studia appositamente per le automobili in modo da rendere possibili ed accessibili servizi di pubblica utilità e servizi al conducente (come gli aggiornamenti sul traffico, il suggerimento di percorsi alternativi, l'aggiornamento delle mappe satellitari, ecc) e per i contenuti multimediali (come la radio) così da eliminare una volta per tutte il supporto fisico all'interno della plancia del veicolo.
Per eliminare definitivamente i supporti l'idea di fondo è di allacciare tutto alla rete tramite wireless. Si tratta sempre di una rete multi-tecnologia spinta nella sua evoluzione dai contenuti multimediali. I servizi di mobilità delle automobili saranno sviluppati in maniera indipendente da ogni casa automobilistica (rendendo così difficile che automobili di marche diverse comunichino tra loro).
Uno schema di questo tipo è abbastanza folle perchè a ben pensarci esistono già le reti cellulari disponibili con punti d'accesso ogni 300m tant'è che varrebbe la pena di utilizzare UMTS per ogni cosa, tuttavia sfruttando il wireless non si pagherebbero gli operatori telefonici per utilizzare il servizio che verrebbe erogato a chiunque a prescindere dal proprio operatore telefonico e dalla zona di copertura (che potrebbe appartenere ad un altro operatore). Quanto appena ipotizzato potrebbe accadere domani, ma in un ipotetico dopodomani sarà ovvio dover rinunciare alle reti cablate anche e soprattutto per una questione di manutenzione che, nell'etere, non serve (abbattendo molto i costi): si parlerà di costruire una enorme rete ad hoc (→ rete ad hoc: non esistono veri e propri access point e le schede di rete comunicano tra loro senza bisogno della presenza di una data infrastruttura) su tutta la città. Aumenteranno le reti di tipo *opportunistico* dove si sfruttano le connessioni create dagli utenti tra di loro per compiere altri tipi di traffico dati. Si vorrà dare anche connettività in cielo sugli aerei (cosa che una compagnia di volo australiana già fa ma, siccome in Australia la connettività è molto costosa, la compagnia aerea non solo la fa pagare in supplemento al biglietto ma non mette a disposizione dei viaggiatori le prese di corrente così da scoraggiarne l'utilizzo e limitare il traffico). Oggi in aereo si spegne il telefonino perchè i vecchi aeroplani non sono schermati per le onde elettromagnetiche (e cambiare un'intera flotta è improponibile), ma

accade anche in quelli più recenti che sono comunque schermati: si tratta di una strategia per evitare telefonate perchè in volo una telefonata può essere comunque fatta ma l'operatore non riesce a tariffarla per via della velocità con cui ci si aggancia e ci si sgancia dalle celle.
Sono dette **reti di sensori** quelle reti costituite da dispositivi piccolissimi che vengono spesso sparpagliati all'interno di una determinata area per compiere delle rilevazioni ambientali/strutturali i quali dispongono di una batteria ad induzione. Li si possono immergere nel cemento e se avvertono qualche variazione di stabilità o altro mandano un segnale. Quelli che sfruttano la luce possono essere utilizzati come degli "occhi" sintetici, utili da impiegare nella videosorveglianza. Esistono branche che si occupano anche delle reti di sensori multimediali.

Wireless su scala geografica

Il wireless su scala geografica e non più prettamente cittadina ha sue specifiche caratteristiche ed anomalie. Il backbone non è più un problema significativo perchè un provider ha generalmente abbondanza di risorse. Le difficoltà sono spesso tecnologiche ed organizzative come l'eterogeneità delle tecnologie (che sono spesso complesse specialmente sul backbone quando deve attraversare paesi con regolamentazioni e tensioni d'alimentazione differenti e discordanti), l'allocazione dello spettro (difficile per via della sua natura frammentaria, inoltre ogni paese ha le sue regole interne); anche se si dovesse scendere a compromessi risolvendo i due problemi sopra citati si avrebbe comunque difficoltà a rendere possibile una corretta e funzionale comunicazione (handoff) tra tecnologie differenti.



Esempio. *Service provider in USA → esistono un sacco di operatori telefonici differenti negli Stati Uniti e che utilizzano tecnologie molto diverse tra loro. Lo spettro allora è un colabrodo per via della quantità di interferenze che queste tecnologie operano tra loro (anche se spesso tentano di sovrapporsi senza*

creare troppo scompiglio). A livello energetico si può parlare di tecnologie più o meno performanti:

1. *Cingular (Nationwide GSM/GPRS/EDGE system);*
2. *T-Mobile USA (GSM/GPRS/EDGE 1900 MHz voice and data network);*
3. *Verizon Wireless (AMPS,CDMA20001xRTT,EV-DOsystems);*
4. *Sprint Nextel (CDMA (Sprint PCS) and iDEN (after Sprint-Nextel Merger), 1xRTT, EV-DO systems);*
5. *Alltel (CDMA,AMPS,EV-DO).*

Lo schema/grafico di cui sopra indica l'allocazione dello spettro (non di tutto ma solo di una piccola fetta) per quanto riguarda gli Stati Uniti: mostra come i vari segnali e le varie tecnologie si contengono e dividono una certa banda di frequenza.

*La **banda ISM** (Industrial, Scientific and Medical) costituisce una specie di "buco" identico per tutti i paesi del mondo all'interno del quale le frequenze non sono allocate in maniera statica a qualche servizio che paga per utilizzarle e sono messe a disposizione di chiunque le voglia utilizzare per scopi pubblici o privati, di ricerca o di produzione. All'interno di questa banda si può lavorare con qualsiasi tipo di risorsa wireless ed eventualmente se ne possono implementare di nuove a patto che trasmettano solo in questa banda (in maniera totalmente gratuita). Il guaio è che questa banda contiene una quantità mostruosa di interferenza perchè è la banda dei forni a microonde (non schermati), dei radio amatori (non filtrati) e di tantissimi altri dispositivi (anche il calore di una giornata soleggiata può disturbare le trasmissioni).*

In Europa esiste una mappa del genere... per ogni stato. Siamo davvero lontani dall'unificare gli sforzi a livello europeo e per ogni stato esistono bande riservate all'esercito ed alla televisione che andrebbero un attimo regolamentate (specialmente in Italia).

Wireless broadband

Non si tratta di una tecnologia ma di un processo evolutivo applicato alle reti. Si parla della banda larga ma nel contesto wireless: non è ad oggi un argomento molto chiaro. Si vuole portare gli operatori verso l'uso di tecnologie compatibili tra loro e verso capacità trasmissive molto più elevate di quelle attuali. Per ogni contesto del wireless oggi esiste una tecnologia apposita (*indoor* si utilizza il wifi, *outdoor* esistono i servizi messi a disposizione dagli operatori telefonici): si vorrebbe ottenere una sola tecnologia per tutti i servizi applicativi. Gli operatori telefonici sono innanzitutto aziende che influiscono poiché investono discrete somme in ricerca in direzione spesso diverse e contrastanti per una questione di concorrenza sul mercato. Il WiMax è forse il tentativo più eccellente in questa direzione (è una tecnologia di tipo IP). Oggi quando si acquista un telefono l'operatore telefonico tenta di farci credere che sia il piano tariffario il nodo focale "regalandoci" secondo lui il telefonino, in realtà è il telefono a costituire il grosso della spesa.

Ma cos'è esattamente il BROADband?? Si tratta di *banda*, di un certo tipo, con una certa predisposizione all'utilizzo di UMTS per via dell'ipotetica libertà delle risorse (forse) ma anche qui tutti hanno idee confuse (persino Motorola) su cosa sia realmente il broadband wifi. Il concetto si sfuma naturalmente perchè in verità nessuno ha tutto questo bisogno di banda come ci fanno credere.

WiMax. Acronimo di *Worldwide Interoperability for Microwave Access*, è un

insieme di standard per la fornitura sull'ultimo miglio di connettività alternativa all'ADSL. La sua filosofia di fondo è simile a quella di Ethernet ed è imparentato con il wirelss wifi. Gli apparati dovrebbero essere in grado di fornire una connettività di 40Mbps su ogni canale in un raggio tra i 3 ed i 10 km.
Il WiMax è costituito da diverse generazioni di tecnologie consecutive le quali hanno cambiato lo spettro per poter spendere così meno energia; all'inizio era necessario vedere fisicamente la destinazione del segnale ma oggi non è più necessario; il bit rate è cambiato diminuendo un po' (per non bruciare i piccioni a causa della potenza del segnale in onde elettromagnetiche, è comunque lo Stato che regolamenta e decide la pericolosità o meno di un dispositivo di emissione elettromagnetica); la copertura è rimasta sempre la stessa nel tempo variando in una media di 5 km.

	802.16	802.16a	802.16-2004	802.16e-2005
Date Completed	December 2001	January 2003	June 2004	December 2005
Spectrum	10-66 GHz	< 11 GHz	< 11 GHz	< 6 GHz
Operation	LOS	Non-LOS	Non-LOS	Non-LOS and Mobile
Bit Rate	32-134 Mbps	Up to 75 Mbps	Up to 75 Mbps	Up to 15 Mbps
Cell Radius	1-3 miles	3-5 miles	3-5 miles	1-3 miles

La posatura di questa tecnologia è allo stesso livello del wireless cittadino: con un backbone wireless per fornire connettività ed accesso alla rete Internet attraverso un provider tradizionale (cablato). Anche i terminali si sono evoluti da pc fissi a client embedded con tecnologia WiMax integrata (ma che hanno spopolato poco a causa dell'alto costo dell'integrazione), così si è passati alla produzione di antenne portatili da acquistare a parte ad un prezzo accessibile, in questa maniera è garantita la piena mobilità.

Capitolo 12
I Media in Digitale

I Media in Digitale *(seminario di Mirko Bove)*

Per *digitalizzazione* s'intende l'encoding per lo streaming web. L'**encoding** è il processo di trasformazione di un video analogico in un file video. Esistono diverse tecniche per fare l'encoding:

1. INTRAframe, allora si converte in uno specifico segnale ogni frame di un video preso appunto frame dopo frame, un frame alla volta;
2. INTERframe, dove si tende a diminuire il bitrate del prodotto/video finale prendendo un intervallo in mezzo a due frame precisi detti k-frame ed interpolando tutti i frame che si trovano tra di loro. La qualità cambia a seconda della quantità di k-frame presenti nel video, numero che può variare a seconda di diversi fattori (a pesare sulla quantità, oltre alla quantità di k-frame sta anche la dimensione dello spazio che intercorre tra loro).

Formato e *codec* sono due concetti diversi. Il formato è una sorta di contenitore nel quale si inserisce poi il codec. Il video analogico non possiede uno standard unico ed universale vantando una quantità di parametri spesso incompatibili, similmente possiede queste caratteristiche anche il video digitale il quale possiede compatibilità differenti (accentuate da HTML5): si tenta comunque di dare una copertura massima per tutti i differenti client.
Lo streaming web richiede la produzione di diverse versioni della stessa risorsa così da venire in contro e cercare di soddisfare le richieste di tutti i vari client tra loro diversificati. La qualità del video digitale dipende anche dal *bitrate* e dalla *dimensione della finestra* (di visualizzazione), in sostanza dipende dal codec. L'encoding per il web richiede l'ottenimento della migliore copertura per tutti i browser maggiormente utilizzati (quindi per tutti i client, e bisogna rendersi conto che non si tratta solo di fornire un formato adeguato ma anche una qualità conforme alla connessione che ogni client possiede o meno a sua disposizione). Le eventuali diciture *hd-ready* e *full-hd* sono eredità provenienti dal mondo del video analogico (in digitale non hanno particolare significato) e servono per specificare le dimensioni della finestra. Molto importanti sono le proporzioni del video (16:9, 4:3, ecc); nel video analogico i pixel non sono quadrati allora quando si passa al video digitale esistono dei sistemi abbastanza complessi che calcolano i valori di conversione. L'intraframe è più adatto per video contenenti moltissimo movimento e scene molto rapide; con l'interframe si utilizzano codec con un bitrate variabile che all'inizio viene fornito come valore medio ma che poi può essere modificato a seconda della natura del video: nelle parti di video in cui compaiono scene rapide si aumenta strategicamente il bitrate, diversamente lo si abbassa dove non c'è necessità di seguire spostamenti veloci.
Nel *master* vengono stoccati i file master, ovvero quei video in qualità massima utilizzati spesso nella post produzione, tipicamente prodotti direttamente dalla videocamera; la post produzione non può essere fatta su video di scarsa qualità oppure su video ottimizzati per la rete per via del fatto che non è affatto facile distinguerne nettamente i frame.
Il Centro Televisivo Universitario (CTU) dell'Università degli Studi di Milano

possiede un enorme archivio di video analogici tutti salvati su vecchi supporti: si ha spesso a che fare con master video divenuti oramai illeggibili a causa della vecchiaia dei supporti ormai non più leggibili (perchè la tecnologia non lo permette o perchè rovinati). Per recuperare e tentare di salvare questa enorme mole di dati si sceglie spesso di digitalizzarla nella speranza così di ottenere e conservare video in alta qualità (appositamente recuperati in alta qualità perchè non è detto che un domani anche il web sia pronto a ricevere video di questo tipo codificati in maniera da non perdere nulla della bellezza del video originale e non sarebbe possibile adattare a questi scopi video in partenza già di bassa qualità). Esistono delle regole per la digitalizzazione:

1. innanzitutto si può sfruttare il **time code** per separare diverse registrazioni che possono trovarsi sullo stesso supporto (poiché bisogna riservare un file a ciascuna registrazione come se si trattasse di un contenuto atomico);
2. si tolgono i **neri** e le **barre colori** che una volta servivano per la taratura dei dispositivi di visualizzazione;
3. ecc.

Nel CTU di Milano si è scelto di utilizzare il formato .avi per la digitalizzazione dei video con un alto bitrate. Si pone tuttavia il problema di *dove* mettere tutta questa mole di dati; anche se la digitalizzazione è necessaria (perchè oggi comunque tutti i video sono registrati direttamente su file e non più prima sulla cassetta) non si sa poi dove stoccare tutte le informazioni. Così vanno individuate proprietà fondamentali e ferree per effetuare l'immagazzinamento delle registrazioni individuando delle specifiche politiche: si possono utilizzare una serie di dischi ottimizzati per l'immagazzinamento collegati da una serie di ponti sia a livello fisico che sul web (sul web è più lento ma lì non servono dischi performanti e speciali collegamenti tra di loro); bisogna anche pianificare lo spazio sul disco necessario che però non dipende dalla larghezza della finestra quanto dal bitrate (che determina la dimensione finale del file).
Digitalizzare ed immagazzinare i file non garantisce la loro durata eterna, così vanno individuate delle tecniche per il backup dei dati (storage ridondato; dischi di backup fatti in maniera periodica che tuttavia occupano una certa quantità di risorse sia in termini di calcolo che di tempo macchina; esistono anche politiche per migliorare il numero di bitrate di un file come la **deduplica** che per non replicare blocchi di byte uguali si tende a creare dei riferimenti a blocchi già backuppati in precedenza così da ridurre fino a 10 volte la dimensione del file; cloud ma che non è per forza la soluzione migliore, bisogna studiare quanto viene a costare mantenere attivo ed utilizzare il servizio). Nemmeno lo spazio per lo storage è infinito così si sceglie di archiviare direttamente su un unico nastro una quantità di n registrazioni svuotando di conseguenza l'archivio dei video master (ma sul web non si fa la stessa cosa poiché se così si facesse si toglierebbe dalla rete il file e nessun utente potrebbe più scaricarlo e guardarlo). Le cassette di backup vanno ripristinate spesso perchè non sono eterne e col passare del tempo tendono a deteriorarsi. Più backup si eseguono, meglio è: può sempre capitare un disastro e compromettere così parte dell'archivio il quale, se duplicato e messo al sicuro a distanza ragionevole, può non subire mai danni.

Infrastruttura hardware

Si parte dal mondo analogico che si suddivide in *girato* (quindi tutti i documenti video importanti a livello storico e culturale), la *produzione* derivata dal girato,

documenti acquisiti e le *apparecchiature* per le dirette/registrazioni live. Si passa poi alla codifica la quale s'interfaccia con un due storage uno per la bassa e l'altro per l'alta qualità (ai quali sono collegati dei dispositivi di backup). Prima di passare alla parte di pubblicazione sul web s'incontra un server/computer che s'interfaccia con il database detto *strumento di backoffice* il quale gestisce la catalogazione dei video con informazioni relative ai dati ed a come sono stati classificati (lo strumento di backoffice spesso interroga il database e gestisce i metadati dei video). Il CTU si è costruito uno strumento di backoffice tutto da solo perchè sarebbe stato oneroso e difficile trovare un software già fatto (o da farsi fare) per gestire i metadati dei loro contenuti che sono particolari e difficili da leggere. Dopo di chè, il video passa direttamente all'erogazione sul web attraverso dei server dedicati allo scopo i quali s'interfacciano al database; c'è anche un server per lo streaming live che s'interfaccia direttamente con lo studio di registrazione.
Il database organizza i contenuti per operare delle catalogazioni e delle ricerche su di essi; compie una *taggatura* sui video in maniera tale da attribuire a ciascuno delle keyword identificative (cosa che può avvenire in diversi modi: tramite il contributo del pubblico che attribuisce una certa classificazione al contenuto, funziona solo su grandi moli di pubblico e per la legge dei grandi numeri alla fine si ottiene un risultato ottimale; chi carica il video compie automaticamente il procedimento di taggatura definendo anche una gerarchia all'interno delle keywork per cui la prima indica l'ambito, la seconda la categoria e la terza può essere la natura del contenuto, così da rendere davvero efficace il procedimento di tag stesso; taggatura automatica in base al testo di descrizione che accompagna il video attraverso un software di analisi del testo che seleziona ed isola le parole più ricorrenti e ne fa dei tag); tiene traccia delle effettive visualizzazioni dei video e da quale browser/client provengono; gestisce gli automatismi su eventuali nuovi formati di codifica e sui nuovi formati video; pubblica in modo organizzato i video master così da renderli reperibili e facilmente consultabili. Lo schema del database si basa sul *progetto* che include in esso tutte le nature della produzione video (master e per il web) dal quale si diramano il filone *master* e quello per il *web*. Sul filone relativo all'erogazione per il web si trovano poi, per ogni produzione, una divisione in *capitoli* che spesso aiuta a gestire ed erogare i contenuti (specialmente quando sono molto complessi e lunghi); ad ogni capitolo è poi riservato uno *stream* diviso a seconda della qualità e del formato del file (solitamente lo stream si compone di 3 formati diversi per ciascuno dei quali si prepara una versione ad alta qualità ed una a bassa qualità a seconda del bitrate deciso, il tutto per rispondere alla maggior quantità di client con esigenze drasticamente differenti). Per accontentare la maggior parte del pubblico è necessario analizzare qual'è il traffico e da quali client proviene, così da specializzare la fornitura di servizi per quel determinato pubblico. Le produzioni più importanti si divulgano su 4 bitrate differenti.

Live Streaming

Spesso lo streaming live lo si fa tramite server dedicato HTTP il quale, in seguito ad una richiesta proveniente da un client, recupera il contenuto e lo manda in download al cliente in modo che lui poi, nel mentre, possa iniziare a goderselo (cosa che fa Youtube). Lo streaming server, in aggiunta a quanto appena detto, può anche compiere un maggior controllo sullo stato del cliente (per esempio su

dove è arrivato a guardare il filmato). Nella parte analogica dello streaming live si trova innanzitutto una regia all'interno della quale si decide quale stream proveniente da quale videocamera può andare in onda (spesso le videocamere sono controllare da remoto perchè non è possibile distaccare troppo personale e compiere delle riprese a distanza dal regista); c'è poi una parte di gestione dei contenuti provenienti da VGA come la proiezione a schermo di un blocco di lucidi che vanno convertiti in PAL; c'è anche una parte audio proveniente dai microfoni, dall'ambiente ed anche da un eventuale computer. Attraverso l'uso del DVCAM recorder si registra lo stream su nastro per sicurezza (backup), dopodichè il tutto va in pasto ad un computer per l'encoding il quale lo codifica e lo manda ai server per la fruizione realtime. Questo pc sarebbe gradito replicarlo per tutti i formati video e le qualità alle quali vogliamo erogare il servizio, ma in mancanza di una quantità di macchine fisse da metter a lavorare si può pensare di mandare un unico stream ad alta qualità ad una macchina in studio la quale poi si occuperà di instradare i vari formati/qualità ad altre macchine locali (il tutto se le macchine sono collegate alla rete). Alla fine viene reso tutto disponibile per gli utenti che faranno richiesta di visione dello stream. Il server di streaming deve assicurarsi di avere abbastanza banda in uscita adeguata rispetto alla richiesta degli utenti (i quali vanno precedentemente stimati).
Il portale video della CTU è sviluppato in Wordpress, prodotto in PHP ed interfacciato ad un database MySQL. Il server di streaming è prodotto con tecnologia Microsoft .NET. Il portale video deve riconoscere automaticamente il client al quale va erogato il giusto contenuto. Non si eroga mai il bitrate massimo consentito dalla connessione poiché bisognerebbe fare delle indagini di complicata realizzazione sulla connettività del client. Quando c'è una live la home cambia proponendo per prima cosa il servizio di streaming live.
Per diminuire i bit di un video si può lavorare con il *frame rate* che può essere abbassato oppure mantenuto per aumentare la qualità del video.

Web Radio

Una web radio solitamente è composta da una parte di automazione, da una di encoding ed una di streaming. La parte di automazione definisce il palinsesto il quale va inviato all'encoder che poi lo converte e lo rende disponibile allo streamer che poi pensa a compiere la divulgazione. Il CTU si affidava ad una serie di tecnologie open source per attuare la propria stazione di web radio: ZaraRadio componeva il palinsesto, JaMin+DarkIce lavoravano sull'audio gestendo i livelli di volume ed IceCast si occupava dello streaming. ZaraRadio poteva fare anche scheduling di una playlist di brani, ritrasmettere altri stream provenienti da altri server. Tra lo streamer/encoder ed l'automazione c'era tutta una quantità di altre lavorazioni del segnale radio.

Capitolo 13
TV Digitale

La TV in Italia

In Italia ha senso parlare, oltre che di TV digitale, anche e soprattutto di TV classica (analogica) perchè tutto incomincia da lì: in Italia infatti esistono e sono sempre esistiti un sacco di canali privati oltre ai pochi nazionali. Fino a 20 anni fa la situazione dei canali privati era senza controllo perchè tra tutti i paesi dell'Europa l'Italia è stata l'ultima a regolamentare le trasmissioni televisive arrivando quando tutti gli altri le avevano già disciplinate dando un freno all'appropriazione da parte di chiunque di qualsiasi frequenza non ancora occupata da nessuna trasmissione (evitando quindi il proliferare delle televisioni private): i privati, infatti, trasmettevano dal proprio garage su una frequenza priva di altre trasmissioni (controllando dove non c'era ancora segnale) e lo facevano più che altro localmente, a livello di quartiere o di paese.
A Milano l'emittente *Telereporter* ha avuto la geniale idea, ad un certo punto (1970), di incrementare le proprie utenze e quindi anche gli introiti derivanti dalla pubblicità inserendo nel proprio palinsesto delle trasmissioni hard → la cosa ha causato una serie di sconvolgimenti a livello di trasmissioni televisive in generale iniziando a danneggiare persino il business della RAI che si vedeva diminuire gli ascolti nell'unica fascia oraria in cui trasmetteva (dalle 4.00 pm alle 12.00 pm). Negli anni '80 nasce il primo network nazionale italiano ad opera di Berlusconi il quale, partendo dall'ambito dell'edilizia (che centrava poco con la televisione) ha tramutato l'emittente *Antenna Nord* (la quale apparteneva alla *Rusconi Editore*) in quella che poi sarebbe stata *Italia Uno* in occasione di un capodanno estendendo per l'occasione la durata delle trasmissioni a tutta l'intera giornata e non più solo alla sera. Chi sovvenzionò la cosa fu un'associazione religiosa la quale era interessata ad inserire nella redazione dell'emittente un certo gruppo di *auditor* (in veste di giornalisti che poi tra le altre cose più avanti se ne andarono/furono costretti ad andarsene). Da qui incominciò ad esistere la televisione su larga scala su tutto il territorio nazionale a partire dagli anni '90 anche se allora era illegale allocare frequenze sull'intero territorio italiano (motivo per il quale Berlusconi andò in contro a numerose citazioni in giudizio). Gli amatori vendettero le loro frequenze di trasmissione così da trasmettere in palinsesto delle cassette registrate dalla grande emittente Italia Uno (in questa maniera si replicava un solo palinsesto in 3 o 4 emittenti che trasmettevano esattamente le stesse cose, a distanza di qualche secondo le une dalle altre così da dare luogo ad un *broadcast* alquanto embrionale). Poi la legge è cambiata rendendo legale quanto prima non lo era. Negli anni '90 le frequenze televisive non erano ancora regolamentate ed erano liberamente allocabili previa richiesta presso gli uffici competenti (con relative trasmissioni gratuite), oggi invece molti dei contenuti trasmessi sulla televisione nazionale sono a pagamento (come certi eventi sportivi, come le partite di calcio o le corse di moto/Formula Uno) per ricalcare un atteggiamento tipicamente europeo già radicato negli altri paesi fuori dall'Italia ma che in Italia è difficilmente condivisibile per via di una certa tradizione relativa alla televisione libera alla quale siamo sempre stati abituati. Negli altri paesi la TV rispetta già una concezione simile a quella di Internet come servizio in abbonamento, per

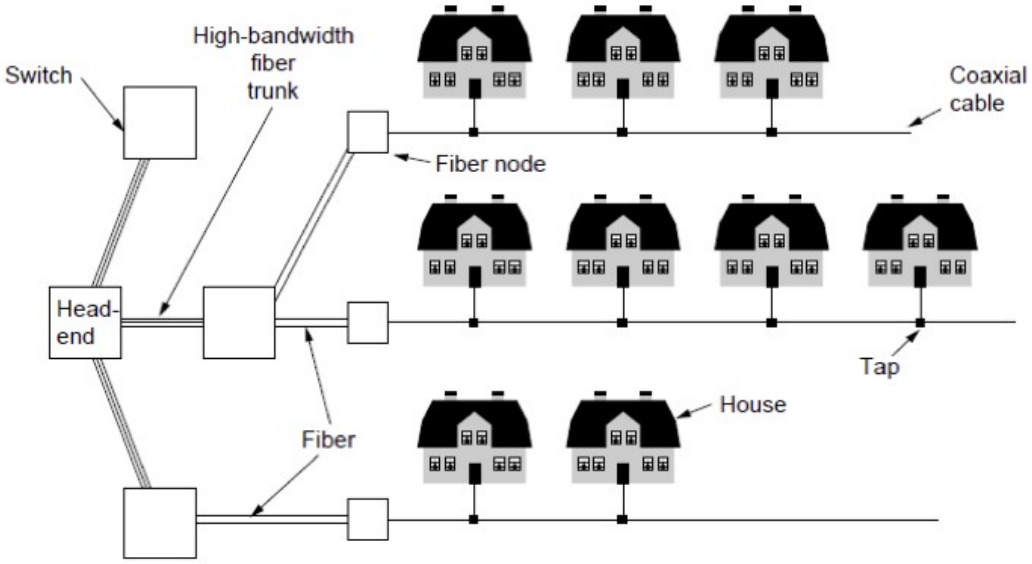
questo motivo trasmissioni come quelle di SKY hanno fatto molta fatica ad entrare nel mercato italiano e Mediaset Premium nei momenti della sua prima comparsa ha dovuto "regalare" molti dei suoi contenuti altrimenti a pagamento. Nel mentre fino al giorno d'oggi si è tentato di passare alla televisione digitale (*digitale terrestre*, tecnologia "nata morta" il cui passaggio è stato vissuto con molta sofferenza: semplicemente nel digitale terrestre la modulazione non è più analogica ma digitale con una conseguente correzione degli errori leggermente più efficiente, e basta), tuttavia la televisione non è ancora interattiva. A proposito di digitale terrestre, in Lombardia sono state allocate più frequenze di quelle disponibili lasciando alcuni clienti paganti senza trasmissione e comprimendo tutte le altre nel poco spazio disponibile così da degradare la qualità delle trasmissioni (in altre regioni la gestione è differente, magari ci sono meno trasmissioni che prendono peggio, ma quelle poche si vedono molto meglio). La tecnologia del digitale terrestre è già obsoleta ed il vero problema con lei è stato il passaggio dall'analogico al digitale.
La televisione è stata un fattore di spinta per l'immigrazione in Italia: molti di coloro che si sono trasferiti nel nostro paese lo hanno fatto poiché avevano imparato la lingua guardando le trasmissioni italiane (spesso si parla di popolazioni che abitano nel bacino mediterraneo).

Televisione Digitale

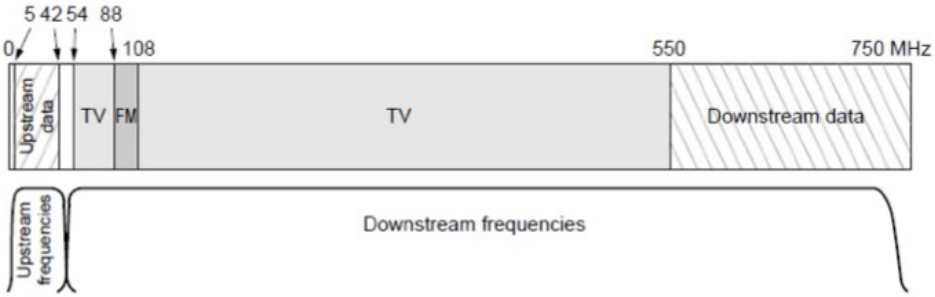
Le trasmissioni televisive digitali costituiscono una evoluzione delle tecnologie analogiche, nate per aggiungere valore aggiunto anche attraverso contenuti personalizzabili (come la possibilità di vedere film interattivi o che possono essere o meno censurati a seconda dell'età dello spettatore tramite dei tag che configurano una visione mirata a secondo del pubblico); c'è inoltre un uso delle risorse più flessibile poiché non le si lega a doppio filo con il destino di una certa trasmissione (la quale, venuta a mancare, rilascia le risorse occupate). Si può classificare la tecnologia digitale in *via cavo*, *satellitare* e *via Internet con IP*.

Cable (TV via cavo). Si utilizza il termine "cable" per indicare la TV via cavo soprattutto nello slang anglosassone. In alcuni paesi è l'unico modo per poter godere di contenuti qualitativamente accettabili e decenti (dove bisogna aver cura di definire il significato della parola "decenti"). Il via cavo ha reso possibile portare anche Internet in molte case sfruttando l'infrastruttura preesistente (e si può scegliere solo tra l'impianto elettrico, quello del gas, il telefono scelto in Europa oppure la TV via cavo preferita negli USA ed in Canada). In Italia infatti si utilizza l'ADSL attraverso il cavo telefonico, non negli USA dove sfruttare la stessa infrastruttura costava troppo così da preferire un sistema basato su HFC via cavo. Il cavo inoltre permette, a livello televisivo, una trasmissione sia analogica che digitale (due tipi di cavo, digitale ed analogico dove sul digitale si tenta un po' di interattività ed un po' di video on demand) con distribuzione di dati in broadcast con un misto tra FDM e TDM. La trasmissione via antenna americana è alquanto scarsa e povera, mentre i contenuti via cavo sono più ricchi e variegati anche se, comunque, non commisurati al prezzo per il quale li vendono (rispetto all'esperienza europea).
Il tutto nasce dalle community per le antenne TV. Mentre in città la ricezione tramite antenna non dava alcun problema spesso negli USA erano i piccoli paesi della campagna a non ricevere bene il segnale televisivo. Si sono presto venuti a creare dei comitati i quali finanziarono la costruzione di una imponente antenna

di ricezione (sovvenzionata alle volte da qualche facoltoso privato) così da creare un espediente per ricevere meglio il segnale che veniva poi inviato tramite un cavo coassiale nelle case di ciascuno degli abitanti di quello specifico paese o di quel certo quartiere (dietro abbonamento). Questo approccio dilagò a tal punto che presto sostituì le ricezioni tradizionali tramite antenna in tutti gli USA.

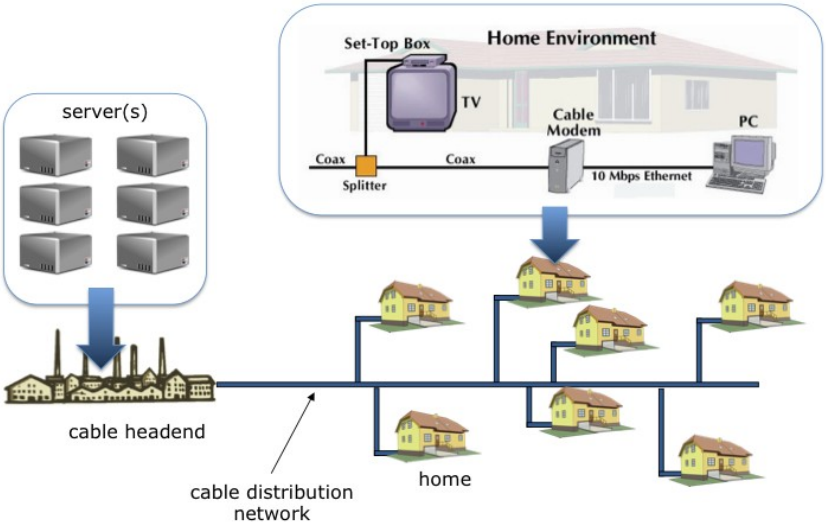


Oggi l'infrastruttura del via cavo è decisamente più complessa di quella delle origini: si parte da un *Head-end* che riceve il segnale dal quale si dipanano diverse direttive in fibra ottica a banda larga le quali sfociano quindi in S*witch* specifici che fanno da nodo centrale ciascuno per un determinato quartiere o gruppo abitativo; dallo switch si passa ancora con la fibra ad un F*iber node* il quale poi si collega alle abitazioni tramite un vecchio cavo coassiale in rame (sul quale esiste una feroce contesa per le risorse, cosa che il DSL risolve eliminando fisicamente il cavo condiviso).



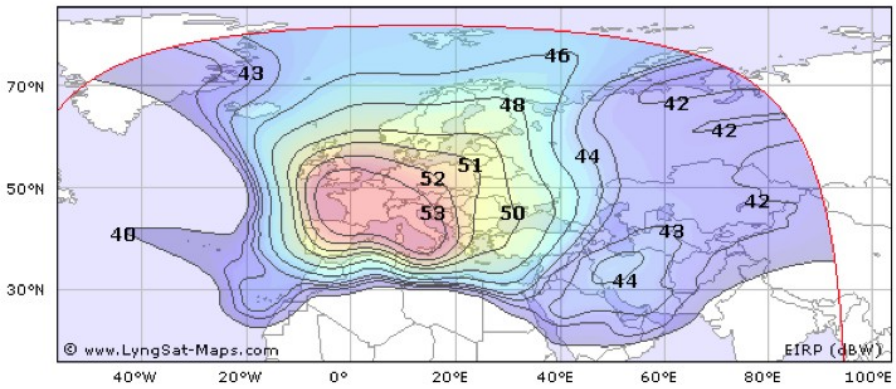
Questo “cavo” viene utilizzato molto non solo per la televisione ma anche per il traffico dati (alle volte anche senza il traffico generato dal segnale televisivo) generando un certo problema riguardo l'allocazione dello spettro oltre che di scalabilità delle risorse per via del canale condiviso. Si deve distinguere il traffico dati da quello televisivo (con DSL sarebbe stato più facile perchè sul mezzo del cavo telefonico il traffico voce è drasticamente poco lasciando grandi margini di manovra a quello puramente dati): negli USA lo spazio dello spettro è occupato praticamente tutto dalle trasmissioni televisive salvo una piccola porzione iniziale ed una finale generando una linea sbilanciata. L'upstream si trova all'inizio nettamente “staccato” dalla porzione televisiva per una questione di sicurezza aggiuntiva mentre il downstream occupa la porzione libera finale

possedendo tuttavia una banda dati limitata dovuta alla tecnologia che, salvo in caso di salto generazionale e di conseguente sostituzione in massa degli apparecchi, non può permettere di alzare la frequenza di trasmissione (quindi non si può aumentare la banda in nessuna maniera).
Una rete via cavo (*cable network*) possiede innanzitutto delle *dorsali/backbone* collegate da un lato alla *cable headend* (chiamata così per tradizione) all'interno della quale ci sono solo una lunga serie di server popolati da dati multimediali direttamente in digitale: attraverso un *gateway* che trasforma il digitale in IP il tutto viene mandato nel backbone il quale distribuisce i contenuti nella rete cablata fino a raggiungere le singole abitazioni; nelle case arriva il cavo coassiale di cui sopra il quale termina all'interno di uno scatolotto/apparecchio chiamato *splitter* il quale separa il segnale dati da quello televisivo (cosa che in Europa è fatta dal filtro DSL): il secondo andrà a finire in un *set-top box* direttamente nella televisione mentre il primo raggiungerà un *cable modem* (equivalente del modem DSL) per essere utilizzato dal computer domestico.



Satellite. Si tratta di un'altra modalità di fruizione della televisione digitale. In molti paesi (con e senza cavo) si è imposto come un nuovo standard di fatto per via degli irrisori costi d'installazione (irrisori solo da parte dell'operatore). Sfrutta la tecnologia dei satelliti geostazionari (e non quella dei LEO che per via delle loro caratteristiche tecniche e delle loro orbite servono solo il servizio di telefonia satellitare) i quali si ricaricano tramite energia solare catturata dagli enormi pannelli di cui dispongono. I costi di lancio delle costellazioni di satelliti sono molto alti ma vanno poi distribuiti sulla quantità di utenti serviti, allora i costi si ammortizzano. Si risolvono i problemi di raggiungibilità degli utenti specialmente in quei paesi con condizioni orografiche svantaggiate ed utenti troppo sparsi sul territorio. La tecnologia satellitare ha tuttavia delle barriere d'accesso oggi insormontabili per i piccoli operatori (anche se a fronte della spesa una volta lanciati in orbita quando non vengono sfruttati possono comunque essere rivenduti) i quali si rivolgono a servizi più banali ed accessibili come Youtube. Il satellite non fa trasmissione on demand ma trasmette (in analogico e digitale) in broadcast con tecnologia FDM.
Un satellite solitamente si alimenta con soli 300 watt (quando l'alimentatore di un qualsiasi computer domestico può raggiungere fino i 450 watt), si posizionano sull'equatore a 36.000 m di altezza con un tempo medio di vita di

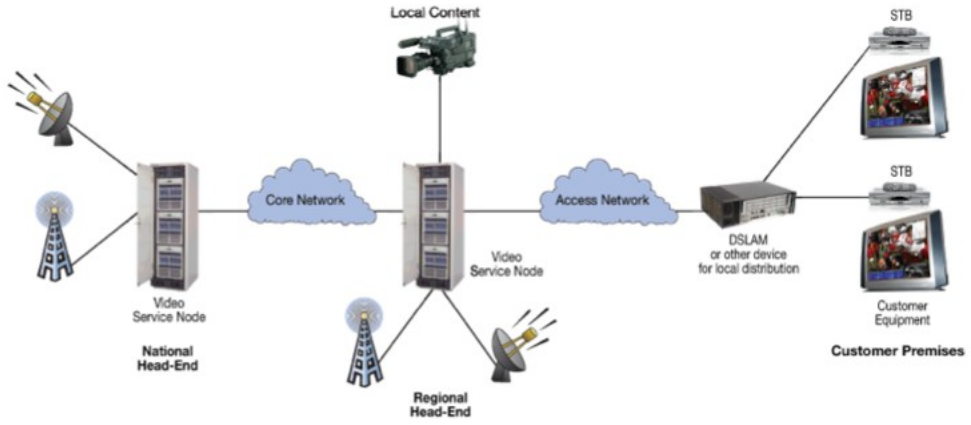
10 anni (chi li manda in orbita solitamente li manda dilazionati così da poterli sostituire un po' alla volta e non tutti assieme). Le uniche note dolenti sono costituite dal Round Trip Time di 250ms il ché vuol dire che tra la trasmissione e la ricezione passa qualche secondo (tra elaborazione, editing, cifratura e gestione del segnale, una volta che viene mandato al satellite ed il satellite lo ridistribuisce si accumula un certo ritardo significativo). Tuttavia la QoS risulta essere uniforme per enormi superfici di distribuzione, in calo solo in certe condizioni meteorologiche e climatiche avverse (brutto tempo).



Hot Bird è un sistema satellitare che copre la maggior parte dell'Europa mediterranea (ma non solo) ed alcuni continenti vicini come Medio Oriente ed nord Africa. In Polonia si usa catturare il segnale di SKY e convertirlo in maniera da riproporlo sul web dietro servizio d'abbonamento, ovviamente molto più economico rispetto a quello originale (cosa altamente illegale). Nelle zone più esterne di una copertura servono antenne paraboliche più ampie per catturare maggior segnale elettromagnetico, tuttavia la copertura risulta essere molto ampia. Siccome tutti ricevono SEMPRE, sorgono spesso problemi legati al copyright di ciascuna zona geografica (bisogna quindi trovare delle strategie di cifratura per selezionare chi può accedere o meno ad un contenuto): se la trasmissione viene fatta tramite Hot Bird si rischia di trasmettere un contenuto per cui non si sono pagati i diritti per la fruizione in un certo stato proprio in quel dato stato (problema che sorge perchè si evita di aggiungere alle trasmissioni un certo grado di cifratura, si dovrebbe pagare di più e si dovrebbero dotare i telespettatori di un apparecchio aggiuntivo per decifrare il segnale): per risolvere il problema semplicemente certe trasmissioni non vengono fatte vie satellite. Le bande satellitari sono molto regolamentate e le trasmissioni sono limitate come l'ampiezza della banda (sennò, trasmettendo ovunque si rischierebbe la sovrapposizione elettromagnetica con altre trasmissioni locali). Si tratta di una comunicazione mono direzionale anche se le antenne per andare in uplink esistono, il guaio è che sono terribilmente costose (caso Eolo in Italia: servizio di connettività che offriva connessione alla rete via satellite, il tutto gestito in arrivo dal satellite ed in uscita dal modem via cavo, con conseguenti casini a livello IP con indirizzi discordanti e costi davvero esorbitanti).

TV via IP. La IP(TV) costituisce il mercato più piccolo di tutti: necessita della presenza costante di una connessione di rete (spesso svincolata dal servizio). Esistono anche le *smart TV* ma non sono esattamente la stessa cosa, con la TV via IP si vorrebbe dare un valore aggiunto alla televisione via connessione

internet. La IP(TV) fa fatica a prendere piede anche per una questione di diritto d'autore che cambia a seconda del paese in cui ci si trova. È molto utilizzata in Canada ed USA anche se non è molto utilizzata in generale per via del target specifico al quale fa riferimento a livello pubblicitario. È costretta ad utilizzare una rete multiservizio; presenta una grande eterogeneità degli utenti tuttavia trasmette sopra IP facendo sia unicast che broadcast che multicast. Youtube non è distante come idea, tuttavia non possiede un concetto di palinsesto (quindi di fruizione vincolata tipica della televisione tradizionale) risultando difficile da fruire per chi non lo conosce bene.
Internet TV eroga lo stesso servizio dell'IP(TV) ma non sono la stessa cosa a livello di QoS (si veda, appunto, Youtube che si basa sulla ricerca mirata del contenuto tramite inserimento da tastiera, improponibile da fare in televisione). IP(TV) è fattibile solo se siamo di fronte ad una rete dedicata che garantisce un'adeguata QoS. IP(TV) è fatta su doppino perchè è così che è fatto l'attacco al modem ADLS, si avrà dunque un DSLAM che eroga solo questo tipo di servizio. Il tutto è migliorabile tramite una rete CDN per IP(TV): si parte da una rete d'accesso con QoS limitato e nella CDN si avranno tutti i server popolati di contenuti multimediali (video) on demand all'interno della quale si ha davvero un effettivo QoS, allora il traffico si fa direttamente da lì.
La televisione comunque non gode solo di contenuti prefabbricati ma anche di dirette, allora la CDN non va più bene; bisognerebbe alimentarla in maniera intelligente per offrire un servizio adeguato, tuttavia non è una cosa facile da gestire.
IP(TV) è interessante perchè è davvero una comunicazione bidirezionale (nella quale è possibile navigare tramite browser); la banda non è regolamentata perchè non esiste il concetto di banda in quanto emittente televisiva (quindi c'è spazio per qualsiasi trasmissione) e si sfruttano infrastrutture preesistenti che possono offrire servizi di valore aggiunto già noti agli utenti (e che quindi verranno sicuramente sfruttati).



IP(TV) si divide in più livelli: una zona di raccolta dati/risorse che colleziona contenuti su ambia scala (a livello nazionale ed internazionale) ed ampia area geografica; del tutto viene fatto un feed e mandato sulla dorsale alla fine della quale l'operatore locale aggiunge al contenuto altri contenuti locali (come pubblicità derivata da una raccolta di contenuti zonali); il tutto viene poi inviato nella rete d'accesso alla quale corrisponde un DSLAM, quindi una o più abitazioni.
I protocolli impiegati nell'IP(TV) non sono diversi dai classici protocolli di rete

utilizzati in internet a livello tecnologico: devono garantire la banda trasmissiva e si occupano di fare streaming di tipo MPEG (spesso via TCP) con marcatori temporali. La vera sfida sta nel raggiungere tutti gli utenti come fa la televisione tradizionale e non come fa DSL, e soprattutto che non richieda l'utilizzo e l'esistenza di un pc (con tastiera e mouse) ma solo ed esclusivamente quello della televisione tradizionale come la intendiamo comunemente.