



il sistema MPEG-2 DVB



**Questo documento è una traduzione
del lavoro pubblicato
sul sito <http://www.salleurl.edu/~si04384/tv/>
dalla facoltà di ingegneria La Salle
dell'università Ramón Llull di Barcellona.**

La presente traduzione non è stata fatta da un professionista né nel campo delle telecomunicazioni né nel campo delle traduzioni.
È estremamente probabile che un tecnico consumato trovi numerose imprecisioni nella terminologia utilizzata
e mi voglio scusare in anticipo anche per i numerosi errori ortografici che sicuramente troverete.
Considerate solo che gli unici moventi per questo 'crimine' contro la terminologia tecnica e la grammatica della lingua italiana
sono stati la buona volontà ed il desiderio di condividere fonti di conoscenza.

| | |
|--|----|
| 1. INTRODUZIONE | 4 |
| 1.1 Storia | 4 |
| 2. MPEG-2 | 7 |
| 2.1. Introduzione allo standard MPEG-2 | 7 |
| 2.1.2. MPEG-2 Audio | 15 |
| 2.2. MPEG-2 Systems | 16 |
| 2.2.1. Introduzione | 16 |
| 2.2.2. Generazione del Transport Stream | 22 |
| 2.2.3. PES packet | 22 |
| 2.2.4. TS packet | 23 |
| 2.2.5. Multiplexaggio | 25 |
| 2.2.6. Modello di Temporizzazione | 28 |
| 2.2.6.1. Introduzione | 28 |
| 2.2.6.2. Program Clock Reference | 30 |
| 2.2.6.3. Timestamps | 32 |
| 2.2.6.4. Presentation TimeStamps | 32 |
| 2.2.6.5. Decoding TimeStamps | 33 |
| 2.2.6.6. Rigenerazione dell' SCR | 34 |
| 2.2.7. Sintassi e semantica del Transport Stream | 35 |
| 2.2.7.1. PES packet | 35 |
| 2.2.7.2. Definizione semantica dei campi del PES packet | 40 |
| 2.2.7.3. Transport Stream Packet | 46 |
| 2.2.7.4. Definizione semantica dei campi di un Transport Stream packet | 47 |
| 2.2.7.5. Campo d'adattamento (Adaptation field) | 49 |
| 2.2.7.6. Definizione semantica dei campi dell'Adaptation field | 50 |
| 2.2.7.7. Transport Streams con più di un programma e con rate variabile. | 55 |
| 2.2.7.8. Trasporto di Program Streams e di Mpeg-1 Systems Streams all'interno di un TS | 55 |
| 2.3. Program Specific Information (PSI) | 57 |
| 2.3.1. Introduzione alla Program Specific Information | 57 |
| 2.3.2. Funzionamento della PSI | 58 |
| 2.3.3. Accesso Aleatorio | 59 |
| 2.3.4. Metodo di trasporto delle tavole PSI | 60 |
| 2.3.5. La "mappatura" delle Sezioni nei Transport Stream Packets | 61 |
| 2.3.6. Program Association Table | 62 |
| 2.3.7. Definizione semantica dei campi della Program Association Section | 63 |
| 2.3.8. Conditional Access Table | 65 |
| 2.3.9. Program Map Table | 66 |
| 2.3.10. Definizione semantica dei campi del Transport Stream Program Map Section | 68 |
| 2.3.11. Sintassi della Private Section | 69 |
| 2.3.12. Definizione semantica dei campi della private section | 70 |
| 2.3.13. Network Information Table | 71 |
| 2.4. Decodificatore (Decoder) | 72 |
| 2.4.1. Transport Stream System Target Decoder | 73 |
| 3. DVB-SI | 77 |
| 3.1 Introduzione al progetto DVB | 77 |
| 3.2. Introduzione alla DVB-Service Information (DVB-SI) | 77 |
| 3.2.1. Concetti e terminologia del DVB | 80 |
| 3.3. Trasporto e Struttura delle tavole DVB-SI | 81 |

| | |
|---|------------|
| 3.4. Tavole DVB-SI | 83 |
| 3.4.1. Network Information Table | 83 |
| 3.4.2. Descriptors della Network Information Table (NIT) | 84 |
| 3.4.3. Bouquet Association Table (BAT) | 86 |
| 3.4.4. Descriptors della Bouquet Association Table (BAT) | 87 |
| 3.4.5. Service Description Table (SDT) information | 89 |
| 3.4.6. Descriptors della Service Description Table (SDT) | 90 |
| 3.4.7. Event Information Table (EIT) information | 92 |
| 3.4.8. EIT Present/Following information | 93 |
| 3.4.9. EIT Schedule Information | 94 |
| 3.4.10. Descriptors della Event Info Table (EIT) | 95 |
| 3.4.11. Time and Date Table (TDT) | 97 |
| 3.4.12. Time Offset Table | 97 |
| 3.4.13. Descriptors della Time Offset Table (TOT) | 98 |
| 3.4.14. Running Status Table (RST) | 99 |
| 3.4.15.- Stuffing Table (ST) | 100 |
| 3.4.16. Altres descriptors | 101 |
| 3.4.17. Sommario della localizzazione delle Sezioni e dei Descriptors | 102 |
| 3.5. Come la Electronic Program Guide utilizza la DVB-SI | 104 |

1. INTRODUZIONE

1.1 Storia

I vantaggi della digitalizzazione dell'informazione hanno coinvolto tutti i sistemi di comunicazione in una sorta di rivoluzione digitale. La copiosità del flusso di bits e la sua capacità di essere immagazzinato e riconvertito, trasmesso e ricevuto, processato e manipolato, il tutto in modo virtuale e senza errori, ne sono la causa principale.

La trasmissione del fax è divenuta digitale fra gli anni 70 e 80. L'audio digitale dei compact discs ha già da tempo sostituito completamente i dischi di vinile e le cassette magnetiche. La TV Digitale è già stata introdotta in un gran numero di paesi, e i dischi video digitali (DVD) stanno entrando in forza nel mercato attuale. Tutto questo grazie anche alla creazione di nuovi standard.

La televisione è stata inventata e standardizzata circa 50 anni orsono e ha continuato imperturbabile ad essere prodotta completamente in analogico a partire dalla telecamera fino al teleschermo nonostante tutti i drammatici cambiamenti tecnologici che si sono susseguiti. Da alcuni anni però, i bassi costi dei circuiti integrati, le reti di comunicazione ad alta velocità, il rapido accesso ai sistemi di immagazzinamento e i nuovi processori, stanno rendendo lo standard analogico obsoleto.

Durante gli anni 80, in piena adolescenza della tecnologia digitale, si sono viste sviluppare le prime normative per la digitalizzazione del segnale video. Il segnale video digitale era conveniente per la compatibilità che offriva fra differenti sistemi, la capacità di non degradarsi anche dopo molteplici copie, la possibilità di creare effetti speciali impossibili da creare in analogico, ed altro. Però presentava anche alcuni inconvenienti: l'elevato flusso di informazione che richiedeva per la sua trasmissione, fra i 108 e i 270 Mbps, rendeva totalmente inattuabile l'implementazione della TV Digitale. Pertanto, rimaneva circoscritta ad ambiti professionali.

La soluzione risiedeva nello sviluppo di tecniche di compressione. Durante gli anni 80 si sono concentrati gli sforzi per il suo sviluppo e soltanto all'inizio degli anni 90 il comitato istituito dalla ISO che prende il nome di Motion Picture Experts Group (MPEG) ha sviluppato il primo standard di compressione digitale comune per l'audio ed il video, la ISO 11172, conosciuto come MPEG-1. L'obiettivo era soddisfare la necessità di immagazzinare audio e video per la prima generazione di CD-ROM, che aveva un bitrate di 1,4 Mbits/s.

Parallelamente si è lavorato alla creazione di un altro standard, compatibile col precedente, però che aveva come obiettivo di specificare una sistema di codifica di audio e video adatto per la trasmissione attraverso differenti reti di comunicazione e per l'immagazzinamento in differenti formati digitali. Il risultato è stato lo standard ISO 13818, chiamato comunemente Mpeg-2.

Benché l' Mpeg-2 contemplasse la compressione dei segnali audio e video, e stabilisse meccanismi di sincronizzazione necessari per poter ricostruire e visualizzare nel ricevitore i differenti segnali originali, non era ancora sufficiente per rendere possibile l'introduzione della TV Digitale.

Nel 1993 si è formato il gruppo di lavoro chiamato Digital Video Broadcasting (DVB), che aveva come obiettivo la definizione di un sistema di televisione digitale per la diffusione via satellite, cavo e terrestre. Questo gruppo ha adottato l' Mpeg-2 come standard di compressione digitale e ha creato un complesso ed esteso insieme di standards chiamato DVB che fra le altre cose definisce l'adattamento del segnale Mpeg-2 a differenti canali di trasmissione (DVB-Satellite, DVB-Terrestre, DVB-Cavo).

Solo a questo punto, con la creazione di questi due standards, l' Mpeg-2 e il DVB si sono aperte le porte alla TV Digitale.

Nel nostro paese (Spagna n.d.t.) la prima piattaforma digitale ha iniziato le sue trasmissioni via satellite verso la fine del 1996, e in quegli stessi anni sono iniziate le emissioni della quarta piattaforma attraverso una rete di diffusione terrestre.



La TV Digitale dal punto di vista tecnico, ottiene miglioramenti sostanziali come:

- Un utilizzo più efficiente dello spettro. Dove prima c'era spazio per un solo canale, ora ce ne stanno sei.
- La codificazione del segnale secondo un solo standard, offre la compatibilità impossibile con i sistemi analogici.
- Più solidità di fronte a difetti ed errori. Si possono eliminare le doppie immagini.

E anche novità irrinunciabili per i telespettatori:

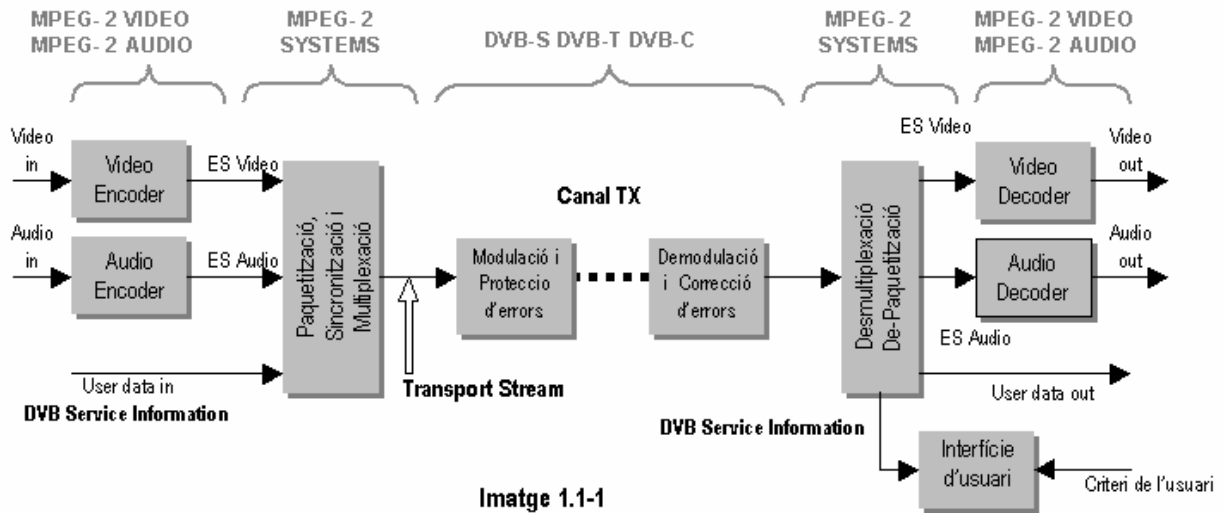
- Aumento del numero dei canali.
- Televisione Interattiva.
- Miglior qualità d'immagine e suono.
- Un passo importante per la convergenza di tutti gli apparecchi audiovisivi.

Però il fattore più importante per il successo della TV Digitale, è, evidentemente, la prospettiva di nuovi affari. È questo l'aspetto che ha veramente fatto la differenza. I settori imprenditoriali sono gli unici capaci di dare impulso ad una implementazione a breve termine, e ciò è garantito grazie alle nuove applicazioni che la TV Digitale permette, che fanno vedere questi settori come delle vere miniere d'oro:

- Televisione a pagamento.
- Pay per view.
- Video on Demand

Gli Standards

Per collocare lo studio di questo progetto introduciamo in forma breve e schematica le aree di operatività dei differenti standards.



2. MPEG-2

2.1. Introduzione allo standard MPEG-2

Nel 1992 il Motion Picture Experts Group, un comitato istituito dalla ISO e formato da persone provenienti da tutte le aree coinvolte (informatica, telecomunicazioni, elettronica, università,...), ha messo a punto lo standard MPEG-1, ufficialmente ISO 11172. Questo standard è stato progettato in modo specifico per applicazioni di immagazzinamento in formato digitale (Digital Storage Media Applications) con un bitrate di 1,4 Mbps.

Durante lo sviluppo dell' MPEG-1, lo stesso comitato, ha constatato che la base delle tecniche che stavano sviluppando era ottimale anche per applicazioni che richiedevano più risoluzione ed un bitrate fino a 10 volte superiore. Ciò ha fatto pensare all'anticipazione dell'implementazione della TV Digitale che si prevedeva allora non sarebbe potuta avvenire che per il millennio successivo, quando si fossero già diffuse le reti a Banda larga. Per cui, allo scopo di unificare i criteri per l'implementazione della TV Digitale lo stesso comitato ha sviluppato quello che oggi conosciamo come lo standard Mpeg-2 (Ufficialmente ISO 13818).

Mpeg-2 doveva stabilire un sistema digitale di codificazione di video e audio, ottimale per la trasmissione attraverso grandi network e per il suo immagazzinamento. È stato approvato definitivamente l' 11 Novembre 1994.

Mpeg-2 è un standard generico. Generico significa che le sue specifiche non sono destinate a una applicazione concreta. Include un insieme di norme che permette il suo utilizzo in una gran varietà di applicazioni e condizioni operative: differenti bitrates, differenti canali di trasmissione o formato d'immagazzinamento, sistemi a ritardo costante o variabile, etc.. Questo significa che l' Mpeg-2 è uno standard flessibile e può essere usato in un gran numero di applicazioni digitali. Da qui la sua complessità.

Per ampliare il suo raggio d'applicazione, Mpeg-2 non stabilisce quali metodi di codifica debbano essere utilizzati, il processo di codificazione, o i dettagli del codificatore e decodificatore. Specifica solo i formati in cui devono essere rappresentati i dati all'entrata del decodificatore e un insieme di regole e processi di decodifica.

Ciò è dovuto alle differenti complessità che possono avere i codificatori a seconda delle necessità delle diverse applicazioni, e quindi, per permettere la continua ottimizzazione e miglioramento dei codificatori. Nonostante sia uno standard completo e definito, la sua applicazione non ha ancora trovato limiti.

L' Mpeg-2 è stato pensato per agevolare l'implementazione delle seguenti applicazioni:

- TV - Per Radiodiffusione Terrestre, Satellite e Cavo.
- HDTV - Per Radiodiffusione Terrestre, Satellite e Cavo. Come il Cinema Elettronico.
- Video in sistemi di immagazzinamento digitale (DSM): CD-ROM, DVD...
- Video nei computers - Video e-mail, sistemi d'informazione multimediale.
- Video on demand (VoD) - Films, eventi in diretta...
- Videocomunicazione - Videoconferenze e multimodalità.
- Network Video - Video in diverse reti di comunicazione: ATM, Ethernet, LAN's...
- HDTV compatibile in SDTV
- Video Professionale - Edizione non lineare, post produzione

L'ampia gamma di applicazioni del Mpeg-2 è sicuramente la chiave del suo successo. Infatti il suo sviluppo ha fatto sì che il campo della multimedialità digitale passasse dall'essere una industria dove ognuno avanzava per conto suo, a una industria dinamica e unificata, con standards che favoriscono l'interoperabilità dei prodotti, e soprattutto, un'industria competitiva.

Lo standard Mpeg-2 è diviso in due parti:

- **13818-1 Systems**

Specifica come combinare o multiplexare differenti tipi d'informazione multimediale in un solo stream affinché possa essere o trasmesso o immagazzinato.

- **13818-2 Video**

Specifica la codifica del segnale video.

- **13818-3 Audio**

Specifica la codifica del segnale audio.

- **13818-4 Conformance**

Specifica come progettare i tests per la verifica dei bitstreams e dei decodificatori.

- **13818-5 Software**

Specifica softwares di simulazione corrispondenti alle parti: Systems, Video e Audio

- **13818-6 Digital Storage Media – Command and Controllo (DSM-CC)**

Specifica protocolli per gestire l'interazione degli utenti con bitstreams MPEG-1 o Mpeg-2 immagazzinati in DSM.

- **13818-7 Non Backward Compatible (NBC) Audio**

Specifica la codifica del segnale audio in un formato non compatibile col sistema MPEG-1.

- **13818-8 10-Bit Video**

Doveva stabilire la codifica del segnale video per un sistema di 10 bits di quantificazione, per applicazioni professionali. Alla fine questa parte è stata eliminata.

- **13818-9 Real Time Interface (RTI)**

Specifica un'interfaccia a Tempo Reale fra l'adattatore del canale di trasmissione e il decodificatore del Transport Stream dei sistemi Mpeg-2. Per applicazioni di telecomunicazione.

- **13818-10 DSM-CC Conformance**

Stabilisce metodi per verificare se una implementazione DSM-CC è conforme alla parte 6 dell'Mpeg-2 : DSM-CC.

Come già anticipato, centreremo il nostro studio sui sistemi Mpeg-2. Per facilitare però la sua comprensione e per meglio definire la sua funzione nel processo di generazione del segnale TV digitale, ci è sembrato opportuno includere anche i due capitoli seguenti, dove si fornirà una una descrizione del processo che subiscono i segnali video e audio prima di giungere all'entrata del codificatore Mpeg-2.

2.1.1. MPEG-2 Video

Le specifiche Mpeg-2 Video partono dal segnale video digitale secondo il formato stabilito con la raccomandazione ITU-R601: 4:2:2 o 4:2:0 , a seconda dell'applicazione.

Il flusso necessario per trasmettere questi formati è di 270 Mbps per il 4:2:2 e di 162 Mbps per il 4:2:0, da ciò la necessità imprescindibile di un sistema di compressione che riducesse questi bitrates così elevati.

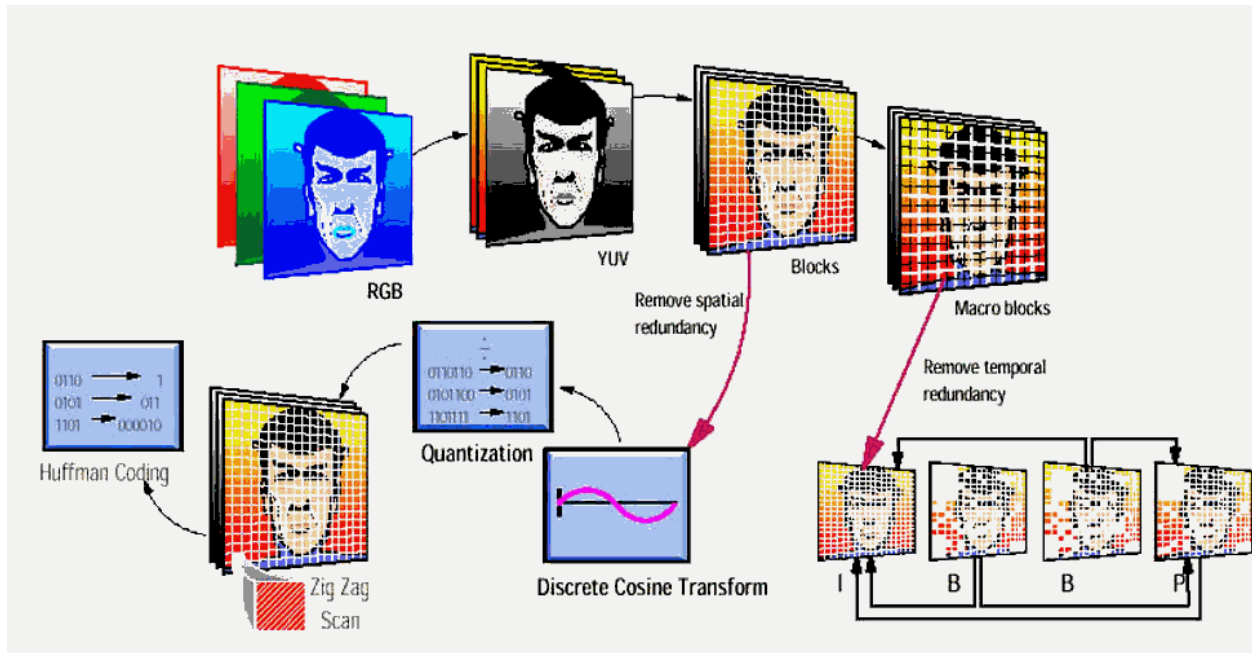


Immagine 2.1.1-1

La compressione del video si basa sulla incapacità che ha l'occhio umano di percepire le variazioni ad alta frequenza, nel segnale a colori.

Si applicano tre tipi di compressione per tentare di ridurre al minimo la presenza, nel segnale, di ridondanza spaziale, temporale e statistica.

La ridondanza spaziale e statistica, sono minimizzate attraverso l'utilizzo congiunto di: Transformata discreta del coseno (DCT), la matrice di quantificazione, la lettura a zig-zag, la codificazione RLC e la codificazione di Huffman (VLC).

Il risultato di questo processo sono le immagini Intra_frames (I-Frames). In queste immagini, la compressione si consegue grazie alla predizione dei valori dei pixels, a partire dai valori dei pixels adiacenti.

La ridondanza temporale è minimizzata con l'utilizzo di metodi di compressione e di compensazione di movimento. Questi riescono a predire il valore di un insieme di pixels di un frame, a partire dalla informazione legata ai frames adiacenti.

Il risultato di minimizzare i tre tipi di ridondanza, sono le immagini Predicted ed i frames Bidirezionali (P,B-frames).

Ripassiamo brevemente i processi utilizzati per eliminare la ridondanza Spaziale, Statistica e Temporale.

Ridondanza Spaziale

Partiamo dai segnali Y,U,V risultanti dal processo di digitalizzazione.

Si divide ogni frame in blocchi di 8x8 pixels per ridurre il tempo di processazione. Si realizza la DCT di ogni Bloc e si ottengono matrici di 64 coefficienti.

Questi coefficienti rappresentano le componenti delle frequenze spaziali, e vengono organizzate nella matrice in modo che: nell'origine (angolo superiore-sinistro) c'è la componente DC, l'asse X indica l'aumento della componente della frequenza orizzontale, e l'asse Y indica l'aumento della componente della frequenza verticale (vedere immagine 2.1.1-2). Questa organizzazione e le caratteristiche della maggior parte delle immagini, fa sì che i valori dei coefficienti diminuisce rapidamente man mano che ci avviciniamo all'angolo inferiore-destro (alte frequenze). Ciò permette la riduzione della ridondanza statistica.

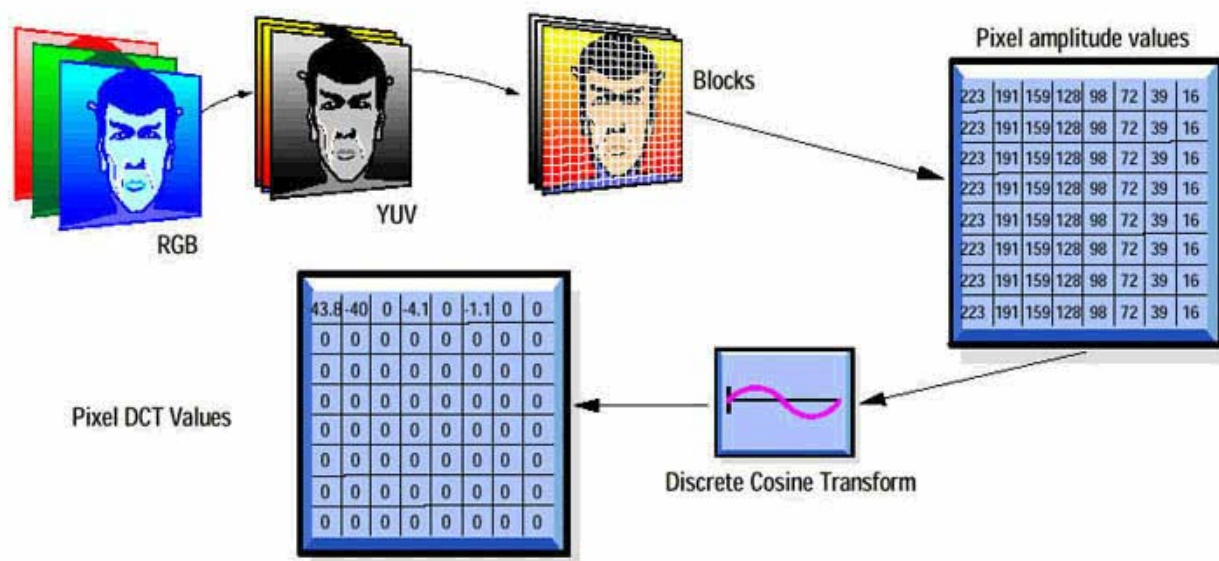


Immagine 2.1.1-2

Nell'esempio, soltanto con l'applicazione della DCT e l'organizzazione dei coefficienti, si passa da un blocco B/N di 8x8 valori (pixels), a 4 valori (coefficienti DCT) diversi da zero.

Ridondanza Statistica

La matrice di coefficienti DCT, viene divisa in una Matrice di quantificazione basata sul comportamento dell'occhio umano (zero per il coefficiente DC, valori elevati per i coefficienti corrispondenti ad alte frequenze). Il risultato è un gran numero di zeri per i coefficienti delle alte frequenze.

L'accumulazione e la distribuzione degli zeri, viene sfruttata (vedere immagine 2.1.1-3) per realizzare uno scanning a Zig-Zag in modo da ottenere il maggior numero di zeri allineati consecutivamente, all'entrata del codificatore RLC. La codifica RLC, sostituisce tutti gli zeri consecutivi, con il numero che indica la quantità di zeri consecutivi: 00000000 \rightarrow 8. Alla fine, il codificatore di Huffman codifica quei coefficienti che si susseguono più frequentemente dal punto di vista statistico, in un numero di bits minore.

Con tutto ciò si consegue una riduzione significativa del bitrate. Però l'utilizzo della Matrice di quantificazione provoca una perdita di risoluzione irreversibile dell'immagine ricostruita.

I valori della Matrice di quantificazione, si possono variare per ottenere una maggiore o minore riduzione del bitrate, però ciò implica una maggiore o minore qualità dell'immagine finale.

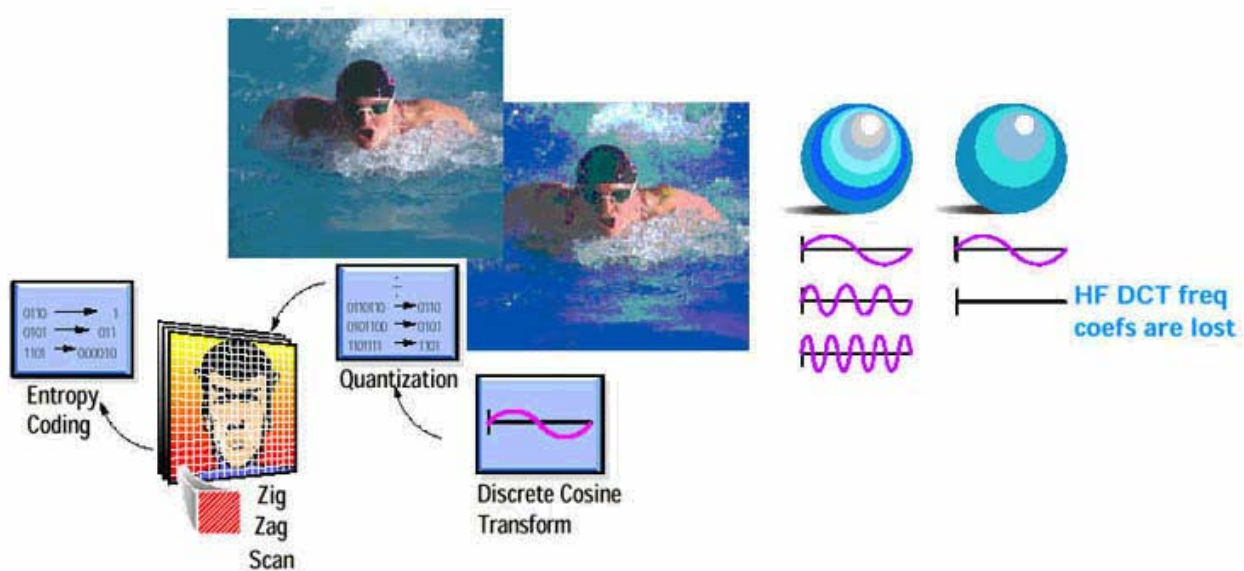


Immagine 2.1.1-3

Nell'esempio riportato nella figura qui sopra, l'immagine superiore non è quantificata, quella inferiore sì.

Giunti a questo punto, dopo aver eliminato la ridondanza Spaziale e quella Statistica, il risultato ottenuto è un I-frame. Se precedentemente si è eliminata la ridondanza Temporale, il risultato sarebbe stato un B o un P-frame.

Ridondanza Temporale

Il primo passo è osservare il seguente frame per vedere le similitudini con l'attuale. Si osserva se il Macrobloc (2 Blocchi di crominanza e 2 di luminanza formano un Macroblocco) situato nella stessa posizione del seguente frame è uguale. A partire da qui si può agire in tre modi diversi:

- Se è uguale, non si fa nessuna codifica. semplicemente si indica che sono uguali.
- Se non è uguale:
 - Nel caso di un P-frame, si controlla se il Macroblocco esiste in una posizione differente nel I o P-frame precedente.
 - Nel caso di un B-frame, si controlla se il Macroblocco esiste nel I o/e P-frame precedenti e successivi.

Se è così, si trasmettono vettori di movimento per indicare il suo spostamento.

- Infine, solo se è un Macrobloc nuovo, viene codificato come qualsiasi Macroblocco di un I-Frame.

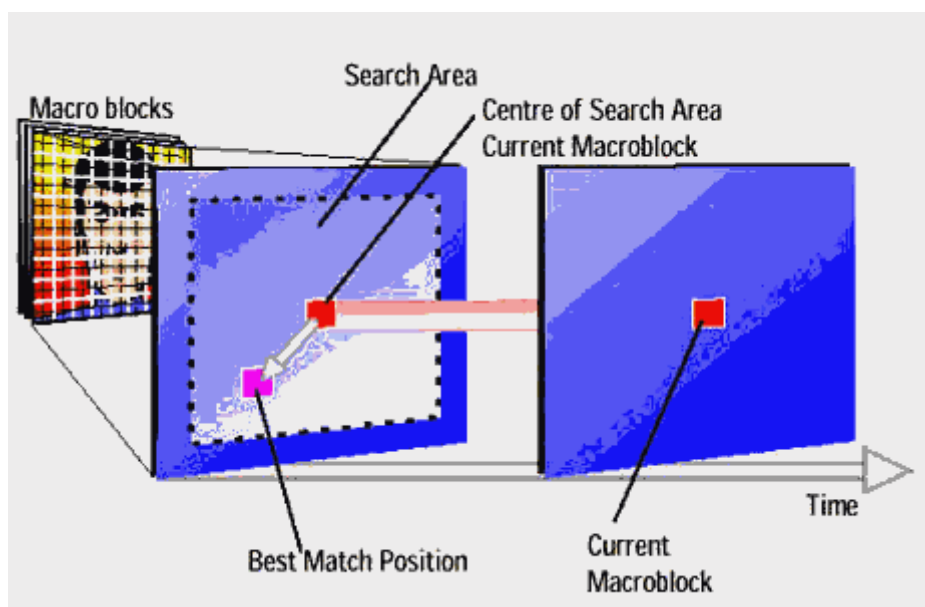


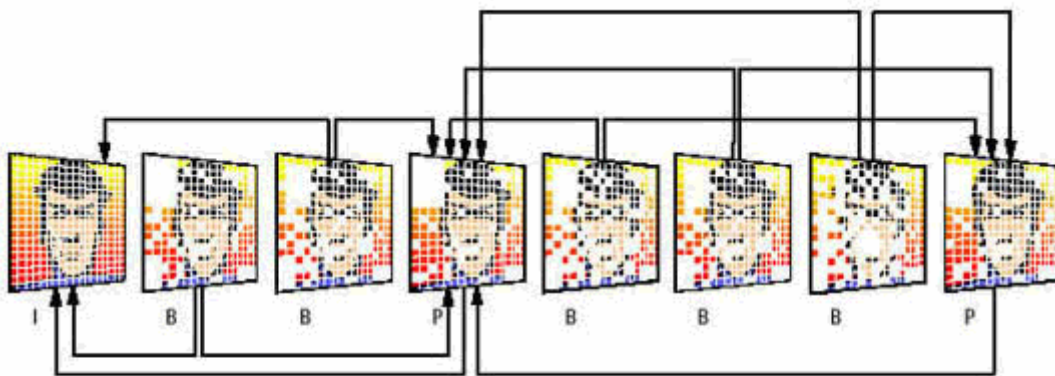
Immagine 2.1.1-4

È In questo processo che si consegue il massimo fattore di compressione, una grande riduzione frame a frame.

Siccome la compensazione di movimento non è perfetta, non si potranno creare lunghe sequenze di immagini nelle quali ognuna dipende dalle immagini precedenti o posteriori, poichè ogni errore si propagherebbe indefinidamente e si amplificherebbe. Da ciò la necessità di tre tipi di immagine I, P e B.

Tre tipi di immagine

- Gli Intra frames sono codificati senza alcun riferimento ad altri frames. Dispongono di tutta l'informazione necessaria per la propria ricostruzione. Sono il punto di partenza obbligatorio per l'accesso a una sequenza. Senza la trasmissione periodica di questi frame, il sistema potrebbe degradarsi rapidamente. Sono le immagini meno compresse. Come minimo ci deve essere un I-frame ogni 12 immagini, per evitare la propagazione indefinita qualsiasi errore.
- I Predicted frames sono predetti a partire da un I o P frame precedente.
- I Bidirectional frames sono quelli che permettono la massima compressione. Sono codificati per interpolazione fra due frames di tipo I o P precedenti e successivi.



- **I-frames: contain full picture information**
- **P-frames: predicted from past I or P frames**
- **B-frames: use past and future I or P frames**
- **Transmit I frames every 12 frames or so.**

Immagine 2.1.1-4

A seconda del tipo di applicazione e della complessità del codificatore si potranno usare: solo I-frames, I e P-frames, o I, P e B-frames. A seconda della concentrazione relativa di ognuno di questi si otterranno fattori di compressione più o meno elevati.

Elementary Stream

Fin qui tutto ciò che abbiamo visto si riferisce alla compressione propriamente detta. Però un decodificatore necessita di informazioni aggiuntive per poter ricostruire i frames in modo adeguato. Per fornire queste informazioni e strutturare i dati secondo una unica sintassi, lo standard Mpeg-2 Video definisce l'Elementary Stream (ES).

Viene stabilita una gerarchia di livelli all'interno di una sequenza:

- *Sequenza*: livello superiore. Determina il contesto in cui si definisce la sequenza (parametri video di base come: le dimensioni dell'immagine, l'aspect ratio (4:3, 16:9), il formato del campionamento, la frequenza d'immagine, escombrat progressivo o interlacciato, il profilo, il livello, il bitrate e le matrici di quantificazione).
- *Gruppo di immagini (GOP)*: livello che permette l'accesso casuale. Inizia sempre con un I-frame.
- *Immagini di tipo I, P, B*: Sono il livello elementare di visualizzazione.
- *Slice*: Insieme di Macroblocchi (normalmente una linea orizzontale).
- *Macroblocchi*: Due blocchi di luminanza e 2 di crominanza (Cr e Ca). Livello dove si realizza la compensazione di movimento.
- *Bloc*: 8 x 8 pixels. Livello dove si esegue la DCT .

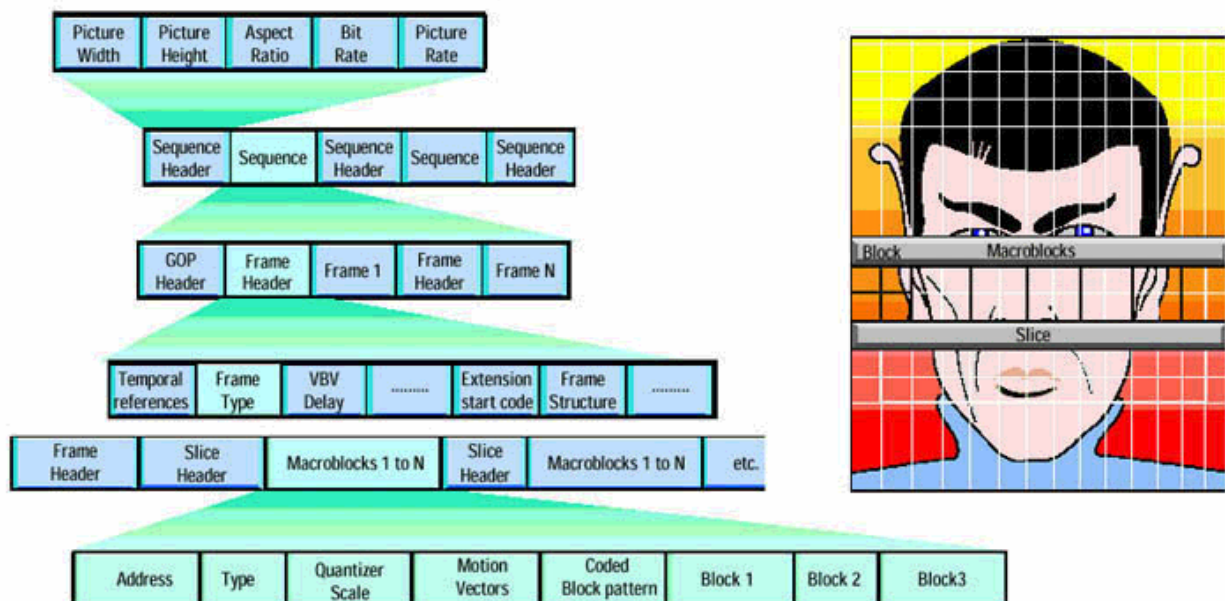


Immagine 2.1.1-5

Nella figura si possono vedere le differenti intestazioni (header) dove risiede l'informazione di sistema.

Questi strati, organizzati nel loro insieme formano un Elementary Stream Video. Pertanto possiamo considerare l'ES, come uno stream continuo di frames compressi, codificati insieme alle informazioni necessarie per la sua ricostruzione.

Però l'ES, non contiene informazioni di temporizzazione né di sincronizzazione, che permettano la ricostruzione della sequenza correttamente temporizzata. È qui che interviene l'MPEG-2 Systems, che fra le altre cose risolve il problema della sincronizzazione, e definisce a partire dall'Elementary Stream, nuove strutture più adatte alla trasmissione e all'immagazzinamento.

2.1.2. MPEG-2 Audio

Due canali audio con qualità CD richiedono un bitrate di 1,4 Mbps. Non è un bitrate esageratamente elevato però, facendo uso di tecniche di compressione arriviamo a 200 Kbps senza pregiudicare significativamente la qualità.

La compressione del segnale audio sfrutta le caratteristiche psicoacustiche dell'orecchio umano, nel quale i toni ad alta potenza tendono a sovrastare i toni di potenza inferiore adiacenti. Il criterio che si applica è "se non lo si può udire, non lo codifico".

Il processo di compressione parte dal segnale audio di frequenza compresa fra 0 Hz e 22 KHz. Questo segnale è diviso in 32 sottobande di frequenza, la subbanda è filtrata, quantificata e codificata secondo le caratteristiche dell'orecchio umano.

MPEG-2 Audio non apporta miglioramenti a livello di compressione rispetto all'MPEG-1 ma apre nuove possibilità, fra le quali quella permettere l'invio di 5 canali audio per conseguire la riproduzione di un suono avvolgente, e fino a 7 canali monofonici per applicazioni multi-lingua.

Esistono due tipi di codifica: una compatibile con l'MPEG-1 Audio e una non compatibile (AC3).

- System relies on masking of tones either side of hi-power tone.
- If you can't hear it, don't code it.
- Sub-band masking takes 3 MBit/s & reduces it to about 200 Kbit/s.
- Two types of audio coding: exists MPEG-2 & Dolby AC3
- US use AC3, DVB uses MPEG-2.

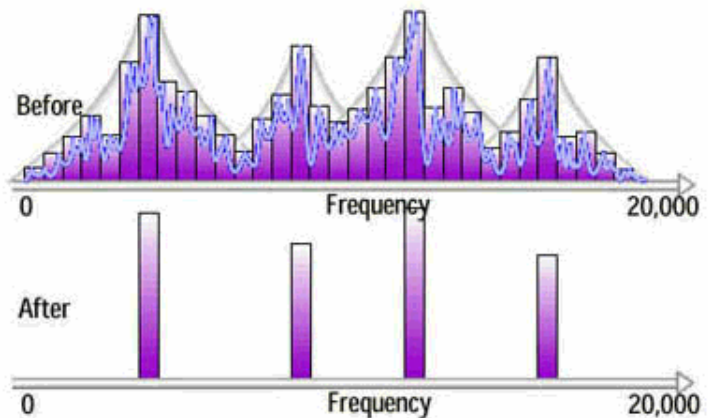


Immagine 2.1.2-1

Il segnale risultante è impacchettato insieme ai dati che contengono informazioni sul processo di codifica utilizzato, per formare un Elementary Stream.

2.2. MPEG-2 Systems

2.2.1. Introduzione

Come abbiamo già visto, l'Mpeg-2 è stato concepito fin dall'inizio come uno standard generico. Uno standard utile per molteplici applicazioni, cioè per molteplici sistemi. Da qui il titolo della sezione Mpeg-2 Systems.

Mpeg-2 Systems specifica la sintassi che fornisce versatilità al sistema Mpeg-2. Fondamentalmente indica come processare gli Elementary Streams per generare uno o più programmi secondo una sintassi ottimale per la trasmissione l'immagazzinaggio.

Un programma in sostanza è un insieme di ES che devono essere visualizzati insieme e in modo sincronizzato.

Requisiti del MPEG-2 Systems

Dai processi di codifica video e audio si ottengono degli Elementary Streams (ES). Un flusso continuo di dati che contiene soltanto informazioni di una sola sorgente audio o video.

È facile intuire che se l'obiettivo è generare programmi e gli ES contengono solo informazione video o audio, sarà necessario qualche sistema per fare delle associazioni fra questi e poter definire, in questo modo, gli elementi costituenti di ogni programma.

Parallelamente, gli ESs che costituiscono un programma dovranno essere sincronizzati. Pertanto, si dovrà anche introdurre qualche sistema di temporizzazione. Un sistema che dovrà risolvere due ulteriori aspetti:

- Fornire informazione sufficiente per riordinare le immagini I, P i B.
Se ricordiamo, gli ESs non contengono alcun tipo di informazione di temporizzazione e, a causa dei metodi predittivi utilizzati, le immagini sono allineate in un ordine differente a quello della sequenza di visualizzazione.
- I ritardi aleatorio introdotti dai differenti canali di trasmissione.

D'altra parte occorrerà risolvere il problema che la natura continua degli ESs non rende agevole la sua combinazione e trasmissione. Pertanto bisognerà definire come si processano gli ES per ottenere strutture più adeguate, e la sintassi di queste strutture.

Da una prospettiva molto generale queste sono i requisiti che la sintassi stabilita dall'Mpeg-2 Systems contempla, e attraverso i quali offre un insieme di regole che permettono infinite soluzioni adattabili a una gran varietà di richieste e applicazioni.

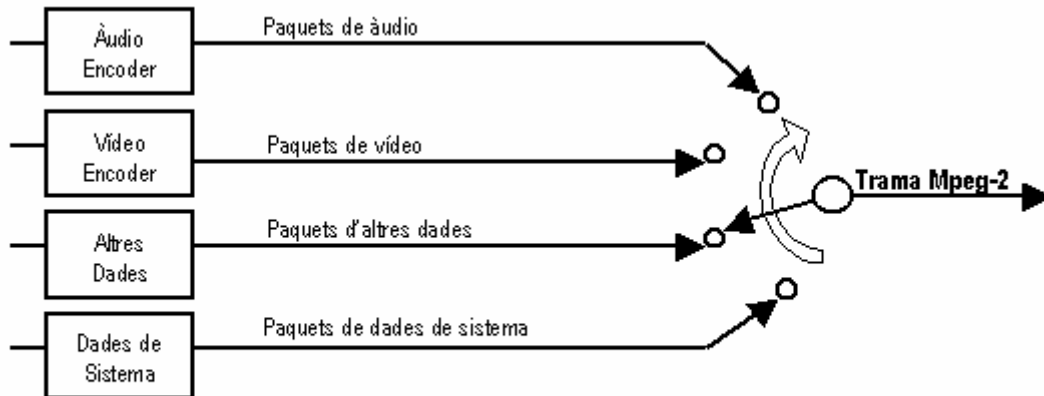
Vediamo ora, come introduzione, il modo in cui l'Mpeg-2 Systems affronta ognuno di questi aspetti.

Multiplexaggio e Strutture

Per potere generare programmi, Mpeg-2 Systems stabilisce come si combinano uno o più ES video e/o audio e altri dati, in uno o più streams appropriati per l'immagazzinamento o la trasmissione. Per realizzare questa combinazione bisogna affrontare la natura continua degli ESs, che come abbiamo già visto precedentemente non permette la sua combinazione e trasmissione attraverso un solo canale di comunicazione. È in questo senso che Mpeg-2 Systems stabilisce un metodo di multiplexaggio, e la sintassi dello stream risultante (multiplexed stream).

In generale, esistono due metodi di multiplexaggio per combinare dati di varie sorgenti in un solo stream. Uno si chiama multiplexaggio per divisione temporale (TDM), i fondamentalmente assegna periodicamente slots di tempo ad ogni elementary stream audio, video, dati, etc. L'Mpeg-2 Systems utilizza l'altro metodo, chiamato multiplexaggio per pacchetti. Con il multiplexaggio per pacchetti, i pacchetti di dati provenienti da differenti elementary streams video, audio o dati, sono intercalati l'uno

dietro l'altro in un solo Mpeg-2 Stream (multiplexed stream), così come si può vedere nella figura 2.2.1-1. Questo tipo di multiplexaggio fa in modo che gli ES possano essere trasmessi sia ad un bitrate costante (CBR) sia ad un bitrate variabile (VBR) semplicemente variando appropriatamente la lunghezza o la frequenza dei pacchetti.

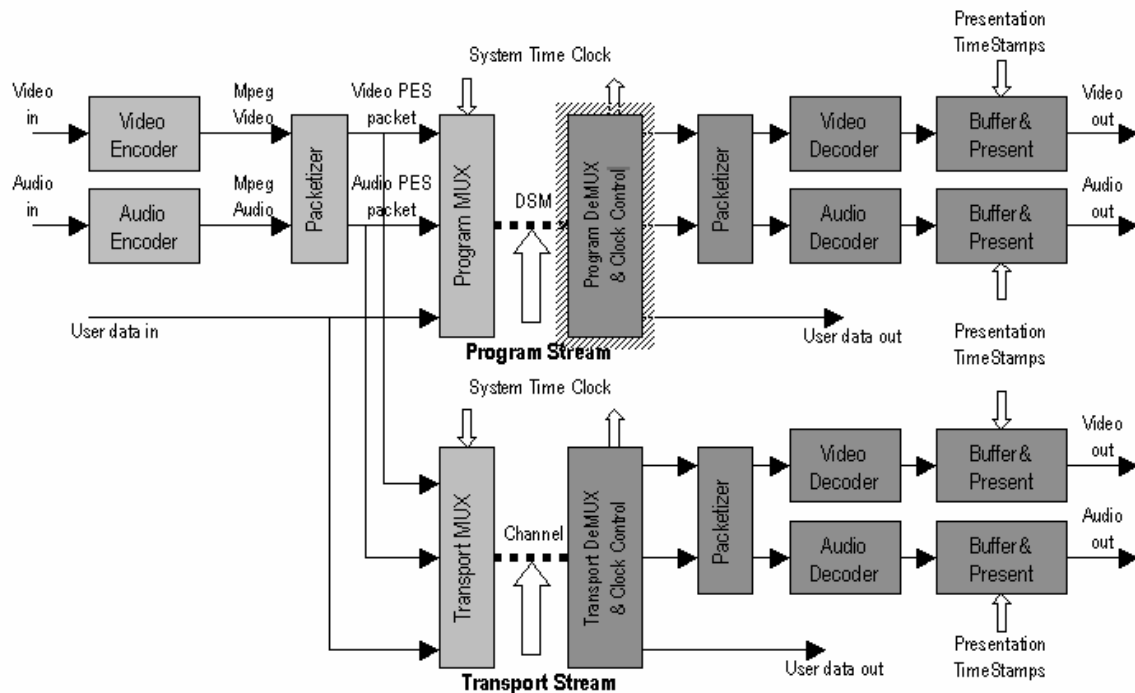


Imatge 2.2.1-1

Al termine del processo di multiplexaggio Mpeg-2 si ottiene uno stream continuo, senza restrizioni di bitrate, variabile o costante, però sempre uguale o superiore al totale dei bitrates degli ES che lo costituiscono.

Mpeg-2 Systems definisce due streams differenti come risultato del processo di multiplexaggio: il Transport Stream e il Program Stream. Ciascuno è ottimizzato per un insieme differente di applicazioni. Il Transport Stream per applicazioni di trasmissione e di immagazzinamento di uno o più programmi, e il Program Stream per applicazioni di trasmissione e di immagazzinamento di un solo programma.

Nella figura sottostante possiamo vedere uno schema di base del processo di generazione e decodificazione di questi due streams.



Imatge 2.2.1-2

Nel processo di generazione il video e l'audio sono codificati rispettivamente secondo l'Mpeg-2 Video e il Mpeg-2 Audio. Gli ES risultanti sono divisi e impacchettati per produrre PES packets. Infine i PES packets vengono combinati con le informazioni di sistema (System Information) per formare un TS (Transport Stream) o un PS (Program Stream). Nel caso di un TS, il multiplexaggio viene eseguito a partire dalla divisione dei PES packets in pacchetti più piccoli chiamati TS packets; al contrario, nel caso di un PS il multiplexaggio viene eseguito partendo da raggruppamenti di vari PES packets che si chiamano Program Stream Packs.

Nel ricevitore, il primo passo è estrarre l'informazione di sistema per poter identificare i dati ricevuti e rigenerare gli ES. Una volta rigenerati vengono decodificati, e le unità di presentazione risultanti da questo processo sono immagazzinate in buffers, dove attendono di essere utilizzate nell'istante indicato dal sistema di temporizzazione.

Program Streams e Transport Streams

Il Program Stream (PS) è simile al MPEG-1 Stream, però utilizza una nuova sintassi in modo da fornire nuove funzionalità mantenendo però la compatibilità con il precedente. Contiene l'equivalente di un solo programma TV e può multiplexare uno o più Elementary Streams con una base di tempo comune. È stato pensato per applicazioni basate su software e con basso livello d'errore come applicazioni multimediali su CD-ROM. Ciò permette di utilizzare pacchetti lunghi e di lunghezza variabile nel processo di multiplexaggio, i PS packs. La lunghezza dei pacchetti va normalmente da un limite di 1 a 2 Kbytes per poter essere contenuta nei settori dei dischi (tipicamente 2Kbytes). Sia come sia, i pacchetti possono avere una lunghezza massima di 64 Kbytes.

Il PS mantiene caratteristiche che erano contenute nel MPEG-1, come il controllo dei buffers del decodificatore, informazioni di temporizzazione, campi per assistere l'analisi del PS dopo un accesso aleatorio e campi per permettere la sincronizzazione degli ES. Però oltre a questo include caratteristiche che non sono supportate dal suo antecessore come il criptaggio dei dati, l'assegnazione di differenti livelli di priorità ai pacchetti, informazioni per assistere l'allineamento dei pacchetti, indicazioni di copyright, indicazione di fast forward, fast reverse, e altri trick modes per dispositivi di immagazzinamento, un campo opzionale per testare il funzionamento del network e una numerazione opzionale delle sequenze di pacchetti.

L'altro tipo di streams definito dall' Mpeg-2 Systems è il Transport Stream (TS), che differisce significativamente dal PS. Il TS può contenere uno o più programmi TV e ognuno con la propria base di tempo indipendente. Il risultato è il multiplexaggio degli ESs che costituiscono i differenti programmi attraverso l'utilizzo di pacchetti di lunghezza fissa, i TS packets, di 188 bytes. La ridotta lunghezza di questi pacchetti è adatta per processi basati su hardware e permette di offrire la solidità necessaria per l'invio di video e audio compresso, attraverso canali disturbati e propensi ad introdurre errori, come le reti di comunicazione tramite cavo e satellite.

Attualmente il TS è progettato per supportare molte funzionalità come il multiplexaggio asincrono di programmi, un accesso rapido al programma desiderato quando si cambia canale, una corretta sincronizzazione degli ESs durante la riproduzione, e il controllo dei buffers del decodificatore durante l'inizializzazione e la riproduzione, sia dei programmi con bitrate costante come di quelli con bitrate variabile.

Le strutture del PS e del TS non seguono strettamente un modello *a livelli*, così è possibile ragionevole convertire l'un nell'altro l'altro, grazie a l'utilizzo comune dei PESps. Non tutti i campi necessari in un PS, sono contenuti in un TS e viceversa, pertanto alcuni devono essere derivati durante il processo di conversione. Sia come sia nessuno di loro è una derivazione dell'altro.

Come già abbiamo detto, gli obiettivi di questo lavoro si centrano sullo studio del Transport Stream, e pertanto non entreranno nelle caratteristiche particolari dei Program Streams. Ciò che faremo sarà comparare alcuni aspetti nei quali le due strutture divergono.

Modello di Temporizzazione

Allo scopo di ottenere una sincronizzazione fra il codificatore e il decodificatore, la presentazione sincronizzata degli ES, il riordino delle immagini I,P i B, e l'annullamento dei ritardi aleatorio introdotti dai network di comunicazione, l'Mpeg-2 Systems stabilisce un Modello di Temporizzazione.

Questo modello si basa sull'inserimento di *timestamps* nello header (intestazione) dei PES packets e di *clock references* nello header dei TS pacchetti. I *timestamps* e i *clock references* sono campioni o valori del clock che interviene nel processo di codificazione e multiplexaggio di ogni programma. Ogni programma ha assegnato un clock che prende il nome di System Time Clock (STC). Questo clock è la base di tempo del programma associato.

In un TS ogni programma ha associato un solo STC, però un STC può essere associato a uno o più programmi contemporaneamente. Al contrario, in un PS, viene trasportato un solo programma e pertanto esiste un solo STC.

Esistono tre tipi di campioni di clock:

- Gli SCR (System Clock Reference) o PCR (Program Clock Reference) che permettono la rigenerazione, nel decodificatore, del clock del riferimento del programma escoltit. PCR è la terminologia che si usa nel TS, poichè non esiste un solo clock di sistema ma un per ogni programma.
- I DTS (Decoding Time Stamp) che indicano in quale istante si deve decodificare ogni immagine.
- I PTS (Presentation Time Stamp) che indicano in quale istante deve essere mostrata ogni immagine o campione audio.

Mpeg-2 definisce soltanto le caratteristiche che devono avere il STC i il PCR, però non stabilisce come devono essere generati ed implementato, lascia questo a discrezione dei produttori.

D'altra parte invece stabilisce la sintassi dei *timestamps* e come devono essere usati. Per spiegare l'utilizzo corretto dei *timestamps* definisce un modello concettuale di decodificatore, il System Target Decoder (STD). Attraverso questo, si spiega in modo teòrico, il funzionamento del sistema di buffers necessario per raggiungere gli obiettivi del Modello di Temporizzazione.

Informazione di Sistema e Definizione dei Contenuti

Per rendere possibile al decodificatore il processo di demultiplexaggio, l'Mpeg-2 Systems definisce per la struttura del TS, la Program Specific Information (PSI).

La PSI è la definizione sintattica di un insieme di quattro tavole e una sezione.

Le tavole PSI sono:

- Program Association Table (PAT)
- Program MAP Table (PMT)
- Conditional Acces Table (CAT)
- Network Information Table (NIT)

La funzione della PSI è fornire al decodificatore le informazioni di sistema: parametri di network, i programmi che compongono il Transport Stream, gli ES che formano ogni programma, la natura di ogni ES, meccanismi per identificare il contenuto di ogni pacchetto, parametri per l'Accesso Condizionato, ed altro.

La sezione si chiama Private Section, e la sua funzione è permettere la trasmissione di dati privati. Cioè, offre al distributore la possibilità di trasmettere qualsiasi tipo d'informazione. Ciò è fondamentale per tutte le applicazioni che possono subire mutazione sull'Mpeg-2 (?) e che richiedono la trasmissione di dati specifici, per esempio: DVB-SI.

Per la struttura Program Stream non è richiesta la trasmissione di così tanta informazione di sistema e pertanto si definisce soltanto la sintassi di un PES packet particolare, il Program Stream Map; questo fondamentalmente si incarica di definire la natura degli ES e le relazioni esistenti fra di essi.

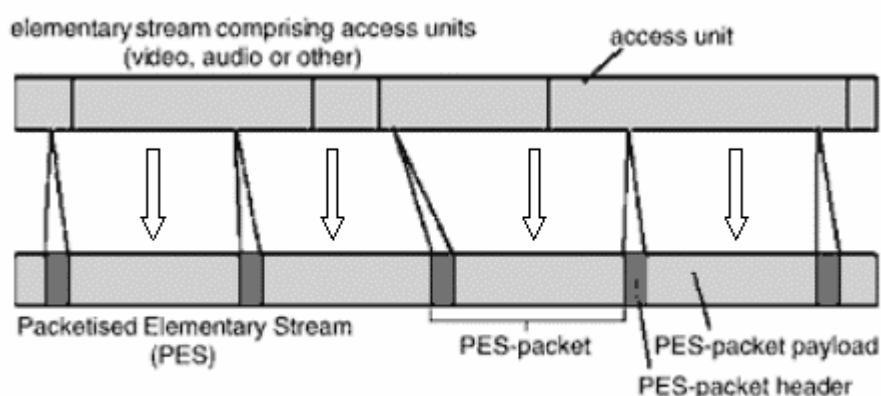
2.2.2. Generazione del Transport Stream

Il punto di partenza per la generazione di un TS sono gli ESs che costituiscono i differenti programmi del TS, insieme ai dati che si desidera aggiungere. Si può anche partire da PS o Streams MPEG-1, però ciò lo vedremo più avanti (punto 2.2.7.1).

2.2.3. PES packet

Il primo passo è affrontare la struttura continua del ES in pacchetti. Questi pacchetti prendono il nome di PES packets. Sono composti da una intestazione (header), la PES header, seguita da un campo di dati (payload o carico utile). La sua lunghezza è variabile e dipenderà dall'applicazione, normalmente sarà di pochi Kbytes.

Il campo di dati è formato da un numero variabile di bytes provenienti da un solo ES e sistemato nello stesso ordine che aveva nella sequenza originale.



Imatge 2.2.2-1

Per poter identificare di quale ES sono i dati che ogni PES packet contiene, la PES header include un campo chiamato Stream_id. Lo Stream_id è un numero di 16 bits che viene assegnato a ogni ES di modo che sia unico e irripetibile nel programma di cui fa parte l' ES, però non ha motivo di essere unico nel TS. Ciò permette di selezionare i PES packets necessari per ricostruire un determinato ES, solamente analizzando l'intestazione dei PESs packets. Lo Stream_id acquisisce differenti valori a seconda del tipo di ES associato (dati, video, audio,...).

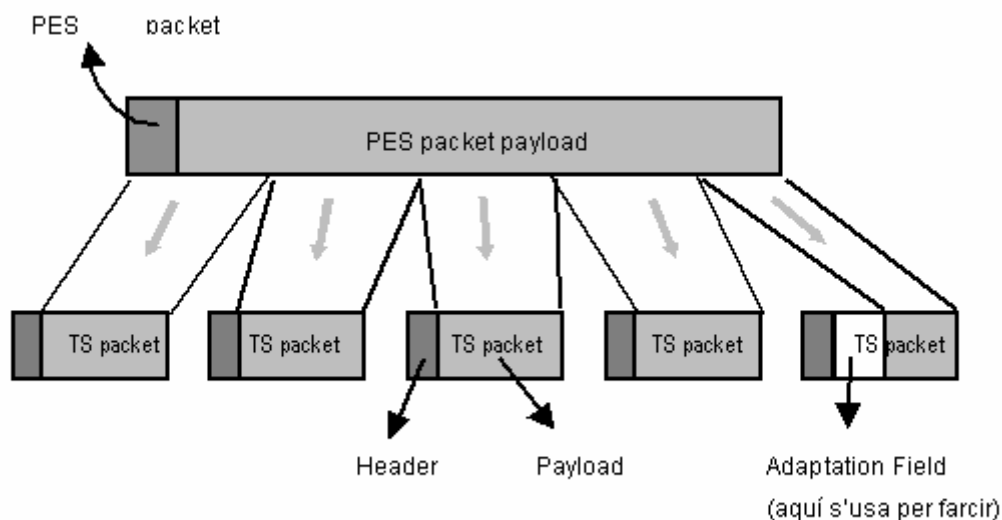
La PES header è di lunghezza variabile, e ha la funzione di fornire le informazioni di temporizzazione e le caratteristiche degli ES che gli stessi ES non forniscono. Se ricordiamo, in un ES abbiamo una sequenza di immagini o campioni audio compressi precedute da una intestazione di sequenza che per il caso del video stabilisce: le dimensioni dell'immagine, l'aspect ratio, il formato del campionamento, la frequenza d'immagine, escombrat (scansione?) progressiva o interlacciata, il profilo, il livello, il bitrate e le matrici di quantificazione. E per il caso dell'audio l'intestazione di sequenza stabilisce: il layer, il tipo di compressione (mono, joint stereo,..) e la frequenza di campionamento. Pertanto la PES header deve trasmettere informazioni sugli istanti di decodifica e presentazione delle immagini dell'audio (PTS e DTS), il trick mode (alta velocità, pausa,..), il sistema di criptaggio, il Bitrate, ed altro che vedremo più approfonditamente al punto 2.2.7.8.

2.2.4. TS packet

Il secondo passo nella generazione di un TS, è sezionare i PES packets e inserirli dentro piccoli pacchetti di lunghezza fissa che prendono il nome di TS packets. La lunghezza totale di questi pacchetti è di 188 bytes. Sono formati da una intestazione (header) di 4 bytes e di un campo d'adattamento (Adaptation field) o un campo di dati (Payload), o da entrambi.

Per questo secondo processo di pacchettizzazione, l'Mpeg-2 Systems stabilisce alcune norme più restrittive al momento di riempire il campo di dati dei TS packets con i bytes provenienti dai PES packets:

- Ogni TS packet può solo contenere dati di un solo PES packet
- Il primo byte di ogni PES packet deve occupare il primo byte del payload di un TS packet.



Imatge 2.2.2-2

Considerando queste due norme, la lunghezza variabile dei PES packets e la lunghezza fissa dei TS packets, diventa evidente la necessità di bytes di riempimento (stuffing bytes) dentro ai TS packets, giacchè difficilmente tutti i PES packets avranno una lunghezza che corrisponda esattamente al payload di un numero intero di TS packets.

In certi casi, per minimizzare l'utilizzo di stuffing bytes, si sfruttano gli stessi bytes per includere informazioni attraverso l'Adaptation field, che è un campo di lunghezza variabile. Però, per non sprecare la capacità del canale, la miglior cosa è fare un calcolo adeguato della lunghezza dei PES packet. Una lunghezza elevata garantisce un uso minimo di stuffing bytes.

Anàlogamente al Stream_id, a livello di TS packet si definisce il Packet Identifier (PID). La sua funzione è indicare a quale ES appartengano i dati trasportati in ogni TS packet. Tutti i TS packets derivati da uno stesso ES contengono nella TS header lo stesso valore di PID. A differenza dello Stream_id, un valore concreto di PID può solo essere assegnato a un solo ES, e deve essere unico e irripetibile nella totalità del TS. Il PID è un campo di 13 bits e pertanto il numero massimo di ES che si possono trasmettere all'interno di un TS è di 8175, tenendo conto che 17 valori di PID sono per usi specifici. Il PID ricopre un ruolo fondamentale nel processo di demultiplexaggio, poiché a partire da esso e dalle tavole PSI, si possono selezionare tutti i TS packets che trasportano gli ES che formano il programma desiderato.

Oltre al campo PID, la TS header è costituita da altri campi che forniscono informazioni a livello di trasporto e multiplexaggio. Alcuni di questi campi sono; il transport priority (indica la priorità del pacchetto), il continuity counter (è un contatore che il codificatore incrementa a ogni TS pacchetto inviato

per indicare l'ordine dei pacchetti aventi lo stesso PID) è l'adaptation field control (indica la presenza o non dell'adaptation field).

L' Adaptation field è un campo opzionale e di lunghezza variabile (da 2 bytes fino alla totalità del pacchetto (184 bytes)). Quando è presente diminuisce la lunghezza del campo dati. La sua funzione è apportare ulteriori informazioni a livello di transport, multiplexaggio e sincronizzazione. Contiene fra l'altro: il campo PCR che trasporta i campioni del STC di ogni programma (imprescindibili per sincronizzare il decodificatore con la base di tempo del programma selezionato) un campo di lunghezza variabile per trasportare dati privati, ed un altro che si incarica dei bytes di riempimento, per quando non ci sono dati per a sufficienza per terminare di riempire l'ultimo TS packet derivato da un PES packet.

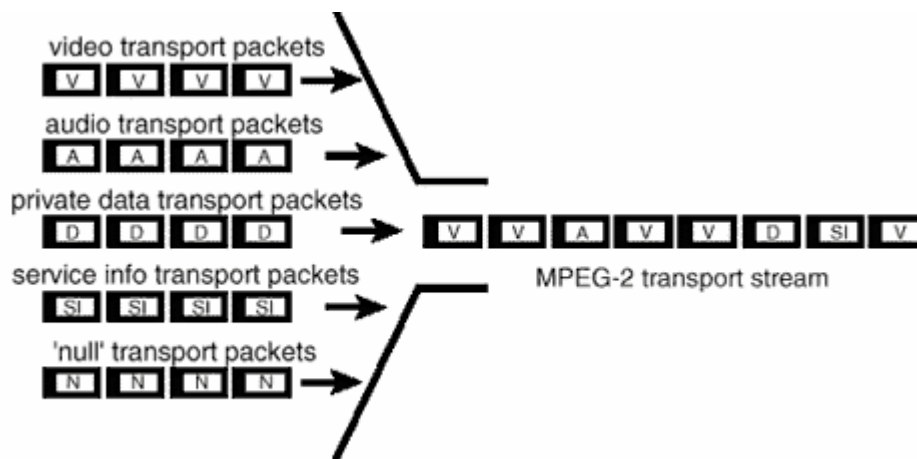


Immagine 2.2.2-2

Ciò che è certo è che la definizione dei PES packets è totalmente concettuale, e un codificatore può generare TS packets direttamente a partire dai ESs di modo che in nessun momento esista uno stream di PES packets (un PES stream secondo l'Mpeg-2 Systems). Sia come sia la struttura del PES packet esiste nel TS ed è imprescindibile.

2.2.5. Multiplexaggio

Infine i TS pacchetti derivati da ogni ES sono multiplexati e allineati per generare un TS. In questo processo di multiplexaggio si combinano anche i TS packets che contengono dati privati o dati PSI.

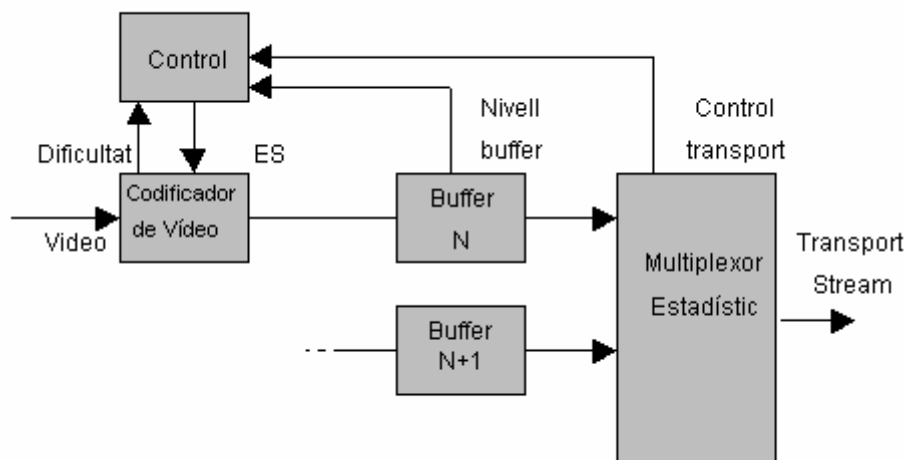
Non sono state definite restrizioni in riferimento all'ordine in cui i TS packets debbano essere multiplexati, eccetto che si deve preservare l'ordine sequenziale fra i pacchetti d'uno stesso ES.

Il bitrate del TS può essere variabile o costante, e come minimo è sempre uguale alla somma di tutti i bit rates degli ES contribuenti.

Il multiplexaggio per pacchetti permette che gli ESs possano anche avere bitrates costanti o variabili, in modo che la frequenza con cui forniscono i TS packets d'uno stesso ES all'interno di un TS, dipende in ogni istante dal bitrate attuale del ES.

In particolare gli ES possono avere un bitrate variabile. Benché il TS abbia un bitrate costante. In questo caso, gli ES che in quell'istante non richiedono un bitrate elevato, possono cedere la propria porzione di canale a favore di quelli che in quello stesso istante richiedono un bitrate più alto. Questa funzionalità si chiama multiplexaggio statistico. Ciò implica l'utilizzo di pacchetti vuoti (null packets) per mantenere il bitrate costante quando le immagini da comprimere non sono particolarmente complicate e il bitrate richiesto è inferiore a quello del TS. Però implica anche che "mai" molteplici ES richiedano nello stesso momento un bitrate superiore al bitrate del TS.

Il meccanismo per mantenere il controllo sul bitrate degli ES si basa su un dialogo fra i blocchi di codifica ed il multiplexer dei TS packets. Nella figura 2.2.2-3 si mostra lo schema di questi blocchi.



Imatge 2.2.2-3

Se si tratta di generare un ES di bit rate costante, questo avrà la sua porzione di bitrate riservata nel TS. Il controllo semplicemente dovrà vigilare che ci sia un livello sufficiente di dati nel buffer per mantenere il bit rate. Se non dovesse essere così, il controllo indicherà al codificatore (encoder) video una compressione più restrittiva (minor qualità d'immagine, si può ottenere modificando la matrice di quantificazione, o utilizzando più o meno immagini B o I). Se al contrario, c'è un livello troppo elevato indicherà una compressione meno restrittiva.

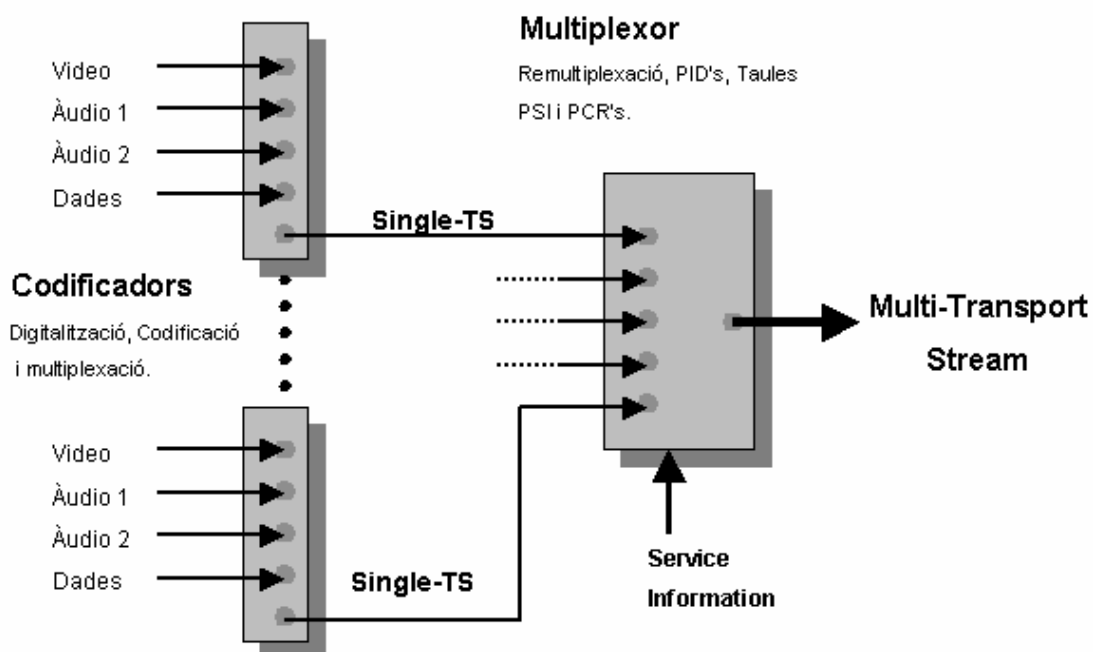
D'altra parte, nel caso di generazione di un TS formato da ESs di bitrate variabile, sarà il multiplexer statistico a gestire il processo. A seconda che un ES richieda più o meno bit rate, il multiplexer varierà la frequenza dei TS packets associati in modo che i buffers non si svuotino mai né si riempiano, e che non si superi il massimo transport rate. Se si tratta di TS di bit rate costante, occorrerà variare la quantità di pacchetti vuoti per mantenerlo costante. E se si tratta di un TS di bitrate variabile, semplicemente si invieranno più o meno pacchetti al secondo.

Nel caso in cui il bit rate richiesto per tutti gli ES superi il massimo transport rate, allora il multiplexer sarà incaricato di avvisare il controllo affinché applichi una compressione più restrittiva agli ESs successivi.

Normalmente un Transport stream richiederà un canale di trasmissione con una capacità di 38 Mbits/seg utili. Conterrà fra quattro e sette programmi, ognuno formato da un video, due o tre canali audio e magari altri dati in formato di sottotitoli, teletext e altro. Ogni Elementary Stream video avrà normalmente bitrates da 4 a 5 Mbits/seg e ogni audio da 128 a 256 kbits/seg.

Remultiplexaggio

In pratica non esistono apparecchi con 20 o 30 ingressi audio e video che generino transports streams con molteplici programmi.



Un sistema per generare un transport stream con più programmi (Multi-Transport Stream, M-TS) sarà suddiviso in due stadi. Il primo si occupa di generare differenti transport streams di un solo programma (Single-Transport Stream, S-TS), e la seconda ha il compito di realizzare una remultiplexazione dei TS packets dei differenti S-TS's (Single Transport Stream), per generare un M-TS (Multiple Transport Stream).

Il primo stadio è formato da un insieme di codificatori Mpeg-2. Ognuno può disporre di vari ingressi per differenti tipi di informazione, per esempio: video, audio, teletext e dati privati. Ogni codificatore (encoder) ha il compito di digitalizzare, comprimere e codificare il segnale video e audio (può anche disporre di qualche sistema di criptaggio), e mettendo insieme il resto dei dati genera un S-TS. Ovviamente ciò include le Tavole PSI ed un PCR.

Tutti questi S-TS's si connettono alle entrate di un Multiplexer Mpeg-2, che forma il secondo stadio. Il Multiplexer ha il compito di rimultiplexare tutti i TS packets dei differenti programmi, per generare un unico Transport Stream, M-TS. Per fare ciò deve modificare le Tavole PSI in modo che riflettano i contenuti del nuovo transport stream, e modificare alcuni valori dei PID, nel caso in cui alcuni S-TS's d'ingresso utilizzino gli stessi valori.

Un'altra funzione molto importante che dovrà realizzare il multiplexer, sarà la modifica dei valori dei campi del Program Clock Reference. Nel processo di rimultiplexazione sarà inevitabile uno spostamento temporale dei pacchetti rispetto alla posizione che occupavano nei S-TS. Ciò, se consideriamo che

concettualmente il PCR indica l'istante in cui il pacchetto associato è uscito dall'encoder, fa sì che i valori dei PCR cessino di essere validi e debbano essere corretti.

Anche il numero di ingressi di un multiplexer è limitato. Pertanto se si vuole creare un M-TS che contenga molti programmi si potrà connettere più di un multiplexer in cascata.

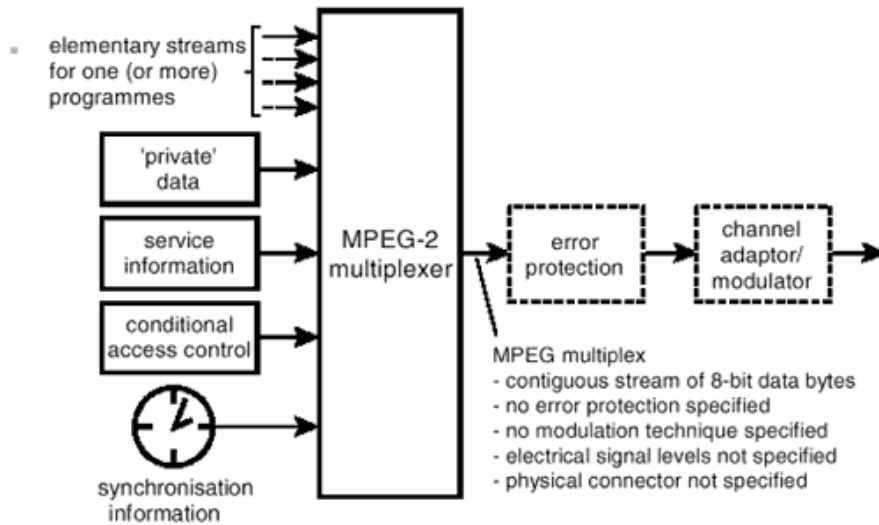


Immagine 2.2.2-4

Le specifiche del Mpeg-2 Systems terminano nel Transport Stream. A partire da qui occorrerà adattare il TS a alla rete ed al canale di trasmissione per il quale si debba trasmettere. Nel caso di network di comunicazione per diffusione, se ne prendono carico gli standards come il DVB in Europa e il ATSC negli USA.

Concretamente il DVB definisce le tecniche di protezione d'errore e il tipo di modulazione necessari per trasmettere il segnale via cavo, satellite e terrestre, così come si riassume nella tavola seguente.

| Parametro | Cavo (DVB-C) | Satellite (DVB-S) | Terrestre (DVB-T) |
|---------------------|--|------------------------------------|-------------------|
| Codifica video | MPEG-2 (MP@MI) MPEG-1 (layer II) DVB-CSA (Common Scrambling Algorithm) 188 bytes (prima del FEC) $1 + X^{14} + X^{15}$ Reed-Solomon (204,188, T=8) Forney, profondità 12 | | |
| Codifica audio | | | |
| Criptaggio | | | |
| Transport packets | | | |
| De-ordinamento | | | |
| Codifica esterna | | | |
| Interlacciamento | | | |
| Codifica interna | Assente | $R_c = 1/2, 2/3, 3/4, 5/6$ o $7/8$ | |
| Fattore roll-off | 15% | 35% | -- |
| Modulazione | QAM 16 a 64 | QPSK | OFDM 2K 8K |
| Ampiezza del canale | 8 MHz (fino a 7) | 27 ~ 36 MHz | 8 MHz (fino a 7) |

Caratteristiche principali dei segnali TV digitali stabilite dalle norme DVB.

Nel caso di reti di comunicazione per commutazione, il TS risulta essere una struttura ottimale per adattarsi al protocollo ATM, poiché i TS packets si adattano perfettamente alla lunghezza dei pacchetti ATM.

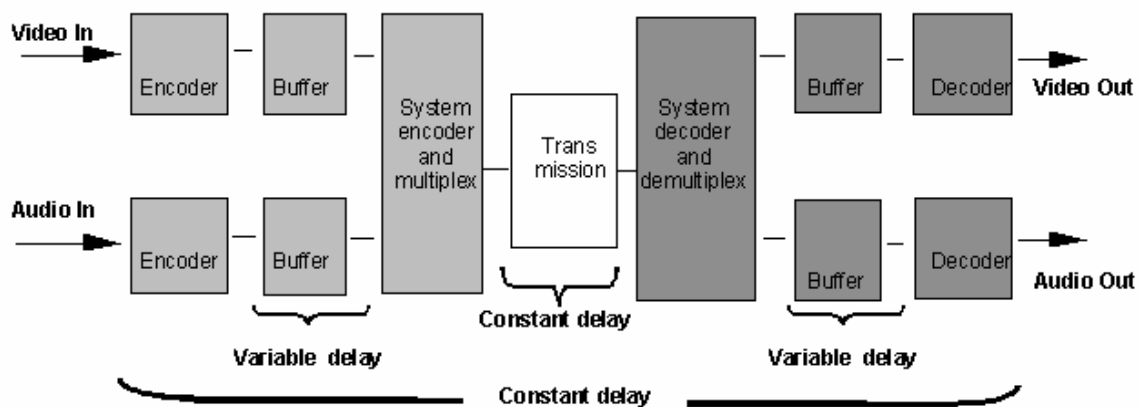
2.2.6. Modello di Temporizzazione

2.2.6.1. Introduzione

L'MPEG-2 Systems raccomanda un modello di temporizzazione che si faccia carico delle specifiche necessarie per ottenere una presentazione sincronizzata dei segnali audio e video che formano uno stesso programma.

Il suo obiettivo principale è ottenere che la sincronizzazione fra le sequenze video e audio risultanti dal processo di decodifica sia identica a quella originale. Cioè, che il tempo esistente fra i campioni audio e video, sia uguale tanto nel codificatore (encoder) come nel decodificatore (decoder). Per ottenere ciò, occorre che ci sia un ritardo costante fra l'entrata del codificatore e l'uscita del decodificatore. È ciò che si chiama ritardo estremo-a-estremo costante.

Per far sì che questo ritardo sia costante, l'Mpeg-2 Systems stabilisce l'utilizzo di buffers (memorie) tant nel codificatore come nel decodificatore. In modo che il modello di temporizzazione si possa schematizzare nel modo seguente:



Imatge 2.2.3-1

Come si può vedere nella figura il ritardo estremo-a-estremo è costante, mentre il ritardo dei buffers del codificatore e decodificatore è variabile. Questi buffers sono indipendenti per gli ES de video e Audio. Per di più, dato che nel codificatore, le frequenze di campionamento audio e video sono significativamente diverse l'una dall'altra, e che la durata di un blocco di campioni audio (una unità di presentazione audio) generalmente non è uguale alla durata di una immagine de video, i ritardi introdotti dai buffers audio e video sono differenti.

Ciò comporta, che la posizione relativa dei dati codificati di audio e video all'interno di un transport stream, non fornisca nessuna informazione di sincronizzazione. Pertanto se gli ES non contengono informazioni di temporizzazione, e l'ordine e la posizione in cui sono trasmessi i dati nemmeno, allora bisogna introdurre nel TS un qualche tipo di informazione supplementare.

È per questa finalità che l'Mpeg-2 Systems stabilisce l'esistenza nel codificatore di un clock di sistema, il System Time Clock (STC). Questo clock gestisce i processi di codifica e multiplexaggio che hanno luogo nel codificatore, in modo che qualsiasi processo che si produce nel codificatore possa essere associato al valore che ha il STC in quello stesso istante. Per cui, in questo modo si riesce a rigenerare nel decodificatore l' STC e si trasmettono i valori del STC associati ad ogni processo, si potrà presentare una sequenza identica a quella originale.

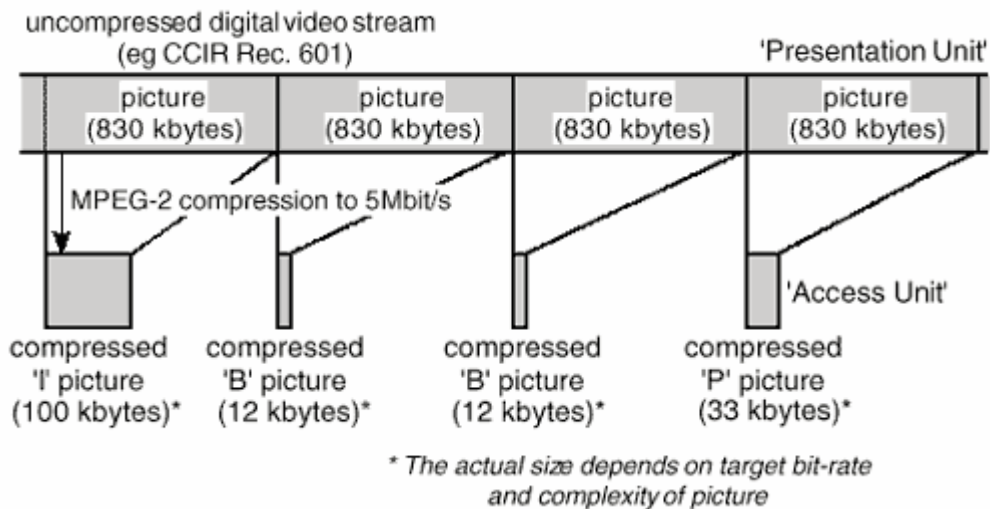


Immagine 2.2.3-2

L'Mpeg-2 Systems chiama i valori del STC **timestamps** (campioni di tempo).

I timestamps indicano la corretta temporizzazione della decodificazione e presentazione dell'audio e del video, così come valori istantanei di riferimento del proprio clock.

I "timestamps" che indicano gli istanti di presentazione di audio e video prendono il nome di **Presentation Time Stamps (PTS)**. Quelli che indicano gli istanti di decodifica prendono il nome di **Decoding Time Stamps (DTS)**. E quelli che indicano valori di riferimento dell' STC e che si usano per la sua rigenerazione nel decodificatore, sono chiamati **Program Clock Reference (PCR)** nei TSs e **System Clock Reference (SCR)** nei PSs.

Questa differenza di nomenclatura fra TS e PS, è dovuta al fatto che il TS può contenere più di un programma e ciascuno di questi associato ad un clock di riferimento indipendente, pertanto in un codificatore TS può esistere più di un STC. Al contrario, un PS contiene solamente un programma e pertanto all'interno di un codificatore PS esiste soltanto un STC.

I PTS e DTS sono trasportati nello header dei pacchetti PES, mentre i PCR sono trasportati nell'Adaptation field dei pacchetti TS.

In definitiva, è la presenza congiunta dell' STC nel codificatore, dei timestamps che sono generati a partire da esso, della rigenerazione dell'STC del programma selezionato nel decodificatore e il corretto utilizzo dei timestamps ciò che permette il controllo intelligente dei buffers, il conseguente raggiungimento di un ritardo estremo-a-estremo costante e la presentazione precisa e sincronizzata dell'audio e del video di uno stesso programma.

Il modello di temporizzazione stabilito dall'MPEG-2 Systems, sotto certi aspetti, è più una raccomandazione che una normativa da osservare. Così, la presenza o non di un STC reale, rimane a discrezione dei produttori, Benché concettualmente debba sempre esistere. Lo stesso vale per la capacità dei buffers e per il suo controllo.

D'altra parte la raccomandazione fatta dall'MPEG-2 Systems è concreta nella descrizione di un decodificatore teorico o concettuale, il System Target Decoder (STC), e concretamente per il TS, il Transport-STC (T-STC). La funzione di questo decodificatore è servire da punto di riferimento per i produttori. Noi analizzeremo il suo funzionamento nel punto 2.4.1 per comprendere meglio il modello di temporizzazione.

2.2.6.2. Program Clock Reference

La base del modello di temporizzazione dell'Mpeg-2 Systems è la rigenerazione nel decodificatore (decoder) del clock di riferimento, l'STC del programma che si vuole decodificare. Se non si ottiene una rigenerazione perfettamente sincronizzata col clock del codificatore, né i PTS, né i DTS, né i buffers, possono essere gestiti correttamente. Pertanto non si può ottenere una presentazione ottimale.

Per ottenere questa rigenerazione, si definisce per il Transport Stream il Program Clock Reference (PCR).

I PCR's sono campioni o valori istantanei del STC che servono da riferimento per la rigenerazione del STC nel decodificatore.

Concettualmente, il PCR è il valore che aveva l'STC, nel preciso istante nel quale il TS packet usciva dal codificatore. Perciò ogni PCR indica il valore corretto che dovrebbe avere l'STC del decodificatore nel momento in cui il PCR viene ricevuto.

Concretamente il PCR è un campo di 42 bits che appartiene all' Adaptation field dei TS packets. La sua trasmissione è opzionale, però deve essere trasmesso **come minimo ogni 100 ms per ogni programma**, per continuare a mantenere la sincronizzazione dei due clock (il DVB stabilisce un intervallo massimo di 40 ms).

Come abbiamo già detto ogni programma ha un sol STC associato, i soltanto i TS packets con PID uguale al valore indicato dal campo PID_PCR, della Program Map Table del programma in questione, possono contenere il campo PCR (vedere punto 2.3.9). Vuol dire che soltanto un ES del programma trasmette i PCR's.

Il campo PCR, ha anche un ruolo molto importante nella determinazione del bitrate del Transport Stream.

Specifiche del STC

L'Mpeg-2 Systems definisce il System Time Clock come un clock governato da una frequenza nominale di 27 Mhz (system_clock_frequency). Questa deve essere mantenuta dentro i seguenti limiti di frequenza:

$$27\ 000\ 000 - 810 \leq \text{system_clock_frequency} \leq 27\ 000\ 000 + 810$$

ed una tolleranza di cambiamento nel tempo $\leq 75 \times 10^{-3}$ Hz/s

Nel punto 2.2.6.6 approfondiremo il metodo con cui avviene la sua rigenerazione nel decodificatore a partire dai PCR's.

Specifiche del PCR

Il PCR è un numero di 42 bits che viene codificato in due parti: la *progam_clock_reference_base*, e la *progam_clock_reference_extension*.

$$PCR(i) = PCR_base(i) \times 300 + PCR_ext(i)$$

Il *progam_clock_reference_base* è un campo di 33 bits espresso in unità di 1/300 del system clock frequency. Cioè, ha una risoluzione di 27 MHz / 300 =90 KHz. Il suo valore è il modulo 2^{33} del risultato intero della seguente equazione:

$$PCR_base(i) = ((system_clock_frequency \times t(i)) / 300)$$

Equazione 2.2.3-1

Dove *i* è un indice per identificare il byte che contiene l'ultimo bit del campo PCR_base, e *t(i)* è l'istante di tempo in cui "teòricamente" questo byte esce dall'encoder (codificatore). Diciamo "teòricament" perchè ciò rimane a discrezione del produttore.

Il *progam_clock_reference_extension* è un campo di 9 bits espresso in unità del system clock frequency. Cioè ha una risoluzione di 27 MHz. Il suo valore è il modulo 300 del risultato intero della seguente equazione:

$$PCR_ext(i) = (system_clock_frequency \times t(i))$$

Equazione 2.2.3-3

Dove *i* è l' indice che identifica il byte che contiene il ultimo bit del campo PCR_ext, e *t(i)* è l'istante di tempo in cui "teòricamente" questo byte esce dal codificatore.

A partire da queste definizioni, il bitrate del Transport Stream è determinato nel decodificatore come: il numero di bytes (del TS) contenuti fra i due bytes che contengono gli ultimi bits di due campi PCR consecutivi dello stesso programma, diviso per la differenza dei tempi indicati da questi stessi PCR's.

$$transport_rate(i) = \frac{((i' - i'') \times system_clock_frequency)}{PCR(i') - PCR(i'')}$$

Equazione 2.2.3-4

i è l'indice di un byte qualsiasi del TS, per $i'' < i < i'$.

i' è l'indice del byte seguente al byte *i*, che contiene l'ultimo bit del campo PCR_base appartenente al programma che si sta decodificando.

i'' è l'indice del byte posteriore al byte *i*, che contiene l'ultimo bit del campo PCR_base appartenente al programma che si sta decodificando.

PCR(i'') è il tempo codificato nei campi PCR_base e PCR_ext in unità del system clock.

La dipendenza di *i* dal *transport_rate* rende evidente la variabilità di questo. Ed il fatto che l'indice *i* si trovi collocato fra $i'' < i < i'$, implica necessariamente che la determinazione del *transport_rate* è posteriore alla sua variazione.

Mpeg-2 Systems stabilisce anche una tolleranza per i valori dei PCR's. La tolleranza PCR è definita come la massima inesattezza permessa nei PCR's ricevuti. Questa inesattezza può essere dovuta alla imprecisione dei valori PCR o alla modifica dei PCR durante un ri-multiplexaggio, però non include gli errori nel tempo di arrivo dovuti alle fluttuazioni del network di comunicazione o ad altre cause. La tolleranza PCR è di ± 500 ns.

2.2.6.3. Timestamps

Prima di entrare appieno nel funzionamento e nelle caratteristiche dei timestamps, è opportuno ripassare la struttura degli ESs.

Un ES de video, benché sia strutturato internamente in Sequenze e Gruppi di Immagini (Groups Of Pictures - GOP), è in definitiva una successione di immagini compresse e codificate. Nella figura che segue possiamo vedere il risultato della compressione di una sequenza di immagini.

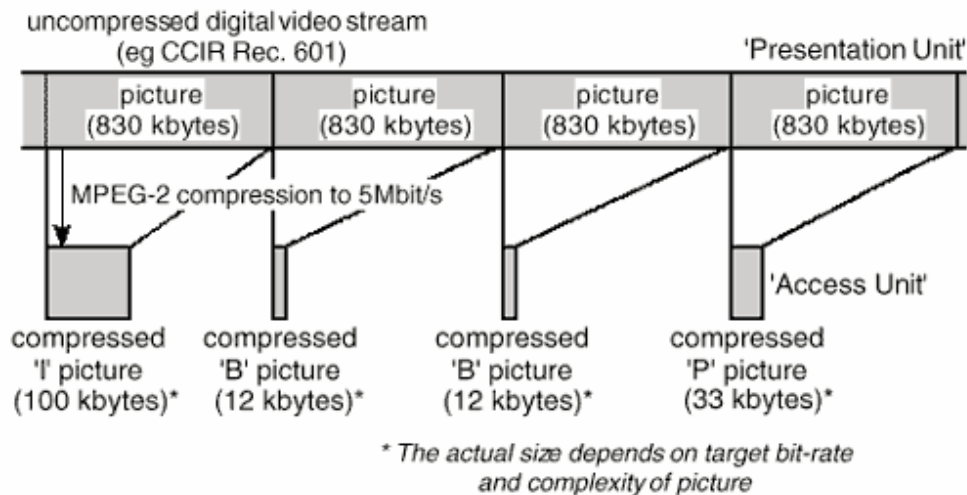


Immagine 2.2.3-3

Le immagini non compresse si chiamano Presentation Units (PU), e l'insieme di bits codificati che rappresentano le PUs, cioè le immagini compresse, sono chiamate Access Units (AU). Per cui, possiamo dire che un ES video è una successione di Access Units.

Come si può vedere nella stessa figura, la lunghezza di ogni Access Unit è differente a seconda che si tratti di una immagine I, P o B, e della complessità dell'immagine.

Per quanto riguarda l'audio, un ES è altrettanto una successione di Access Units. Le Access Units dell'audio sono blocchi audio compresso, normalmente all'incirca di 25 ms di durata.

Boi ci riferiremo ad una Access Unit audio con la sigla AAU, e ad una video con la sigla VAU. Allo stesso modo per le Presentation Units : APU e VPU.

2.2.6.4. Presentation TimeStamps

Le AAU's e le VAU's hanno associati dei Presentation Timestamps (PTS). I PTS's indicano in quale istante, in unità del STC, deve essere presentata ogni PU risultante dalla decodificazione di una AU. Non necessariamente tutte le AU's sono accompagnate da un PTS, come vedremo più avanti a partire da un certo numero di PTS si possono dedurre i rimanenti istanti di presentazione.

L'esistenza di PTS's differenti per l'audio ed il video è necessaria per trasmettere la relazione temporale fra questi due, posto che le PU's di audio e video generalmente hanno durate significativamente differenti e non sono relazionate. Per esempio, una APU di 1152 campioni risultante da un campionamento a 44,1K campioni/secondo ha una durata approssimativa di 26,12 ms, e una VPU con un frame rate di 29,97 Hz ha una durata all'incirca di 33,76 ms. La trasmissione separata dei PTS's audio e video evita la necessità di una relazione stabilita fra la durata e l'intervallo delle APU's e VPU's.

Come vedremo di seguito in molti casi i PTS indicano anche l'istante di decodifica.

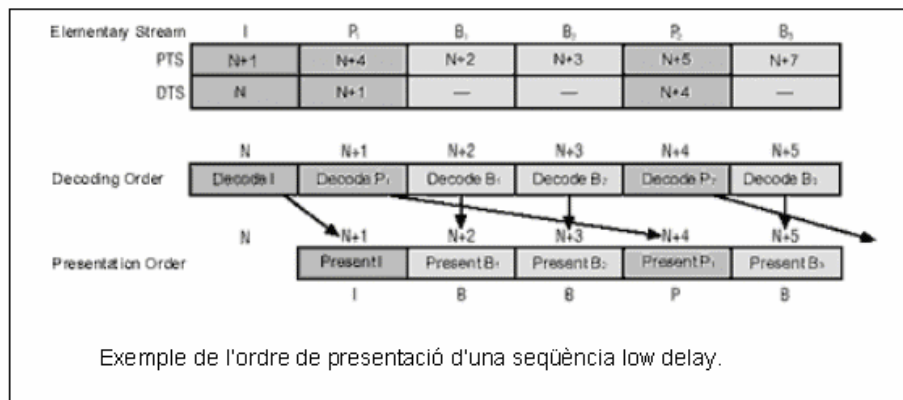
La presenza dei PTS's deve essere **come minimo di uno ogni 700 ms. per ogni ES.**

2.2.6.5. Decoding TimeStamps

Con una funzione simile a quella dei PTS si definiscono anche i Decoding TimeStamps (DTS) che possono anch'essi venir associati ad alcune VAU's (Video Access Units). I DTS indicano l'istante in cui una VAU deve essere estratta dal buffer del decodificatore e decodificata.

La necessità principale dei DTS's sorge dal fatto che negli ES video, le VAU non vengono trasmesse nello stesso ordine che nella sequenza originale, e pertanto, occorre un tempo precedente alla presentazione delle immagini per il riordino delle stesse. Concretamente nelle sequenze formate da immagini B (sequenze non-low_delay), a causa della dipendenza intra-frame che mantengono le immagini codificate, occorre un secondo buffer di ritardo temporale per immagazzinare le immagini I e P, fino al momento in cui possono essere presentate. Il tempo che ogni immagine passa in questo buffer, è la differenza fra gli istanti indicati dal DTS (Decoding TimeStamps) e dal PTS (Presentation TimeStamps).

Per quanto riguarda gli ESs audio e gli ESs video che non richiedono nessun riordino (immagini B, e immagini P e I di sequenze low_delay (sequenze che non sono formate da immagini B)) non è richiesto nessun intervallo di tempo fra la decodificazione e la presentazione, e pertanto l'utilizzo di DTS's è fuori contesto.



Imatge 2.2.3-4

I PTS e DTS hanno la stessa struttura sintattica del program_clock_reference_base. Pertanto, sono campi di 33 bits con una risoluzione di 90 KHz. Ciò significa che non si sfrutta totalmente la risoluzione dell' STC per indicare gli istanti di decodifica e presentazione, però questa risoluzione è sufficiente per tutte le applicazioni per le quali l'Mpeg-2 è stato pensato.

Tutte le AU's possono avere associato un PTS e un DTS, però ogni AU che abbia associato un DTS, dovrà avere anche un PTS. In modo che se sono presenti tutti e due, il DTS indica l'istante di decodifica e il PTS indica l'istante di presentazione (evidentemente il PTS è sempre posteriore al DTS). E se è presente soltanto il PTS, questo indica l'istante di decodificazione e la consecutiva presentazione.

Per le AU's che non hanno nessuno dei due timestamps, il decodificatore deduce il suo PTS per interpolazione. Per esempio, le AU's che non hanno associato il campo PTS e che precedono a una AU che invece lo ha, hanno un istante di decodificazione e presentazione effettivo che risulta dalla somma del DTS (o se non è presente, il PTS) con l'AU precedente, più un valore fisso in unità di 1/300 del STC.

Nel caso di un video codificato a 29,97 Hz, il tempo fra le immagini è di 1/(29,97 Hz) secondi che in unità di 1/300 del STC (90 KHz) equivale a 3003 cicli.

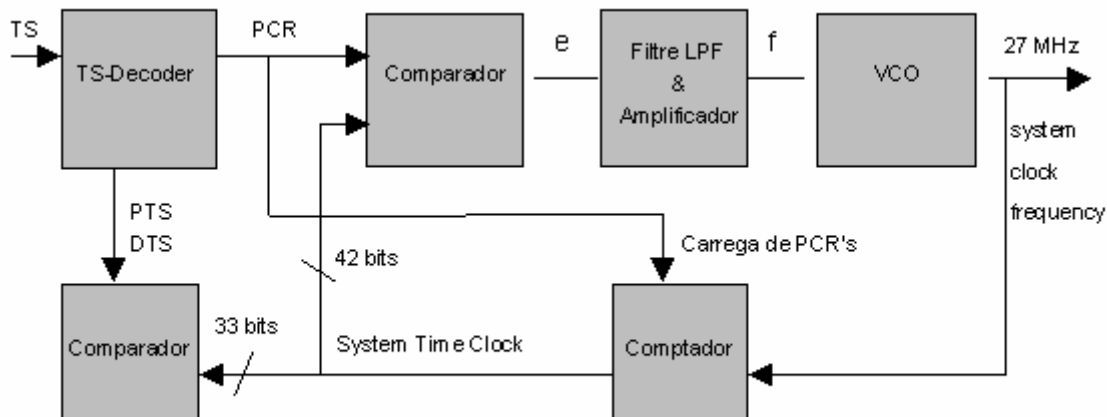
Tempo fra le immagini= 1/(29,97 Hz)=3003 cicli * 1/(90 kHz)

Questi 3003 cicli sarebbero il valore fisso da sommare all'istante di decodifica precedente. La stessa relazione si stabilisce nella decodifica delle successive AUs, Benché il ritardo del riordino incida sulla relazione fra le AUs decodificate e le PUs (Presentation Units) presentate.

Non occorre dire che è imprescindibile che tutti i timestamps associati alle ES che appartengono a uno stesso programma, devono essere riferiti all'STC del programma.

2.2.6.6. Rigenerazione dell' SCR

Per comprendere meglio il ruolo dei PCR's, PTS's i DTS's, esponiamo di seguito un metodo attraverso il quale si può rigenerare l'STC del decodificatore.



Imatge 2.2.3-5

Il metodo si basa su un Phase-Locked Loop, un PLL.

Ogni volta che un telespettatore seleziona un nuovo programma, il decodificatore deve sincronizzare il suo STC con la base di tempo del programma.

A l'inizio dell'acquisizione di una nuova base di tempo si carica direttamente il primo PCR nel contatore, a partire da qui il PLL opera in anello chiuso. L'anello chiuso funziona nel modo seguente. Nell'istante in cui ogni PCR arriva al decodificatore, il suo valore è comparato col valore attuale dell' STC (contatore). Il valore della differenza è linearizzato, e convertito ad un livello di tensione per formare un segnale chiamato "e", il termine d'errore dell'anello. La sequenza di termini "e" viene passata attraverso un filtro LPF ed un amplificatore. La funzione di questi due elementi è regolare la velocità delle fluttuazioni del STC quando l'applicazione lo richiede. Il risultato di ciò è un segnale di controllo, "f", che a seconda del suo valore di tensione modifica la frequenza dell' Oscillatore Controllato in Tensione (VCO: Voltage Controlled Oscillator). L'uscita del VCO è un segnale con una frequenza nominale di 27 MHz: la System clock frequency. Questo segnale si connette ad un contatore che genera i valori STC.

I valori STC hanno la stessa struttura dei PCR, cioè, sono costituiti da due parti che si possono intendere come due contatori: il primo, è di 9 bits, può raggiungere un valore massimo di 300, ed è incrementato con un clock di 27 MHz; il secondo, è di 33 bits, può raggiungere un valore massimo di 2^{33} , viene incrementato con un clock di 90 KHz (ottenuto dalla divisione del segnale di 27 MHz per 300).

I 42 bits dei valori STC vengono utilizzati per realizzare la comparazione con i PCR's ricevuti, e soltanto i 33 bits della base sono utilizzati per realizzare le comparazioni con i valori dei campi PTS e DTS.

Man mano che la frequenza del System Time Clock del decodificatore si sincronizza con quella del codificatore (encoder), il segnale "e" converge verso a un valore costante fino a stabilizzarsi e provocare delle variazioni di frequenza del VCO praticamente nulle. Il tempo iniziale di cui dispone il PLL per stabilizzarsi dipende dalla capacità addizionale del buffer del decodificatore.

In seguito a problemi nel codificatore o alla rete di distribuzione, i campi PCR possono arrivare in anticipo o in ritardo, ciò provocherà piccole oscillazioni nel segnale "e" finché la frequenza del System Time Clock torni a sincronizzarsi. È ciò che si chiama fluttuazioni del PCR (PCR Jitter) che vedremo più avanti. Queste fluttuazioni del STC comportano variazioni nel livello d'occupazione dei buffers e possono finire col provocar il suo svuotamento o sovraccarico, cosa che si rifletterebbe in una presentazione erranea

del programma. Secondo l'Mpeg-2, per un corretto funzionamento del sistema, non è ammesso che in nessun momento i buffers del decodificatore possano soffrire uno svuotamento o un sovraccarico.

Con la frequenza regolata, si può utilizzare qualsiasi PCR corretto per stabilire il valore del STC del decodificatore, e a partire da questo istante, se non esiste PCR Jitter i due STC rimarranno sincronizzati senza la necessità di nessun'altra regolazione. Questa condizione è assicurata finché non sopravviene una discontinuità della base di tempo (per esempio quando appare un indicatore di discontinuità), al che bisognerà ricominciare di nuovo.

2.2.7. Sintassi e semantica del Transport Stream

A questo punto entriamo nel vivo della struttura del Transport Stream. Spiegheremo per prima la struttura dei PES packets e quindi quella dei TS packets. In entrambi i casi definiremo il significato di ciascuno dei campi che li compongono. Questo è il miglior modo per conoscere le caratteristiche e le possibilità che offrono.

Nei due casi, introdurremo la struttura dei pacchetti attraverso algoritmi come il seguente che indica la struttura del TS.

Tavola 2.2.4-1 Transport Stream

| Sintassi |
|-----------------------------------|
| MPEG_transport_stream() { |
| Do { |
| transport_packet() |
| } while (nextbits() == sync_byte) |
| } |

Questi algoritmi facilitano la comprensione della variabilità di questi pacchetti, che non sempre presentano gli stessi campi.

2.2.7.1. PES packet

Trattandosi del primo livello di pacchettizzazione, nei PES packets, alcuni campi hanno una funzione molto vincolata agli Elementary Streams e alla compressione Mpeg-2 Video. Noi abbiamo tentato di esporre in modo chiaro la funzione di questi campi, però sicuramente sarà necessario avere conoscenze a livello di compressione. La funzione di alcuni di questi campi potrà risultare più chiara dopo aver letto il punto 2.4.

Come abbiamo già visto i Transport Stream non soltanto si possono generare a partire da ESs, ma anche possono trasportare Program Streams ed altri TSs. Ciò comporta anche l'esistenza di alcuni campi destinati a facilitare queste operazioni. Come debbano essere utilizzato lo spiegheremo più dettagliatamente nel punto 2.2.7.8.

Tavola 2.2.4-2 PES packet

| Sintassi | Numero di bits |
|--|----------------|
| PES_packet() { | |
| packet_start_code_prefix | 24 |
| stream_id | 8 |
| PES_packet_length | 16 |
| if(stream_id != program_stream_map && stream_id != padding_stream && stream_id != private_stream_2 && stream_id != ECM && stream_id != EMM && stream_id != program_stream_directory && stream_id != DSMCC_stream && stream_id != ITU-T Rec. H.222.1 type E_stream) { | |
| '10' | 2 |
| PES_scrambling_control | 2 |
| PES_priority | 1 |
| data_alignment_indicator | 1 |
| Copyright | 1 |
| Originale_or_copy | 1 |
| PTS_DTS_flags | 2 |
| ESCR_flag | 1 |
| ES_rate_flag | 1 |
| DSM_trick_mode_flag | 1 |
| Additional_copy_info_flag | 1 |
| PES_CRC_flag | 1 |
| PES_extension_flag | 1 |
| PES_header_data_length | 8 |
| if (PTS_DTS_flags == '10') { | |
| '0010' | 4 |
| PTS [32..30] | 3 |
| marker_bit | 1 |
| PTS [29..15] | 15 |
| marker_bit | 1 |
| PTS [14..0] | 15 |
| marker_bit | 1 |
| } | |
| if (PTS_DTS_flags == '11') { | |
| '0011' | 4 |
| PTS [32..30] | 3 |
| marker_bit | 1 |
| PTS [29..15] | 15 |
| marker_bit | 1 |

| | |
|---|-----------|
| PTS [14..0] | 15 |
| marker_bit | 1 |
| '0001' | 4 |
| DTS [32..30] | 3 |
| marker_bit | 1 |
| DTS [29..15] | 15 |
| marker_bit | 1 |
| DTS [14..0] | 15 |
| marker_bit | 1 |
| } | |
| if (ESCR_flag=='1') { | |
| Reserved | 2 |
| ESCR_base[32..30] | 3 |
| marker_bit | 1 |
| ESCR_base[29..15] | 15 |
| marker_bit | 1 |
| ESCR_base[14..0] | 15 |
| marker_bit | 1 |
| ESCR_extension | 9 |
| marker_bit | 1 |
| } | |
| if (ES_rate_flag == '1') { | |
| marker_bit | 1 |
| ES_rate | 22 |
| marker_bit | 1 |
| } | |
| if (DSM_trick_mode_flag == '1') { | |
| Trick_mode_control | 3 |
| if (trick_mode_control == fast_forward) { | |
| field_id | 2 |
| intra_slice_refresh | 1 |
| frequency_truncation | 2 |
| } | |
| Else if (trick_mode_control == slow_motion) { | |
| rep_cntrl | 5 |
| } | |
| Else if (trick_mode_control == freeze_frame) { | |
| field_id | 2 |
| Reserved | 3 |
| } | |
| Else if (trick_mode_control == fast_reverse') { | |
| field_id | 2 |
| intra_slice_refresh | 1 |

| | |
|--|------------|
| frequency_truncation | 2 |
| Else if (trick_mode_control == slow_reverse) { | |
| rep_cntrl | 5 |
| } | |
| Else | |
| Reserved | 5 |
| } | |
| if (additional_copy_info_flag == '1') { | |
| marker_bit | 1 |
| Additional_copy_info | 7 |
| } | |
| if (PES_CRC_flag == '1') { | |
| previous_PES_packet_CRC | 16 |
| } | |
| if (PES_extension_flag == '1') { | |
| PES_private_data_flag | 1 |
| Pack_header_field_flag | 1 |
| Program_packet_sequence_counter_flag | 1 |
| P-STD_buffer_flag | 1 |
| Reserved | 3 |
| PES_extension_flag_2 | 1 |
| if (PES_private_data_flag == '1') { | |
| PES_private_data | 128 |
| } | |
| if (pack_header_field_flag == '1') { | |
| pack_field_length | 8 |
| pack_header() | |
| } | |
| if (program_packet_sequence_counter_flag == '1') { | |
| marker_bit | 1 |
| program_packet_sequence_counter | 7 |
| marker_bit | 1 |
| MPEG1_MPEG2_identifier | 1 |
| originale_stuff_length | 6 |
| } | |
| if (P-STD_buffer_flag == '1') { | |
| '01' | 2 |
| P-STD_buffer_scale | 1 |
| P-STD_buffer_size | 13 |
| } | |
| if (PES_extension_flag_2 == '1') { | |
| marker_bit | 1 |
| PES_extension_field_length | 7 |

| | |
|--|----------|
| for(i=0;i<PES_extension_field_length;i++) { | |
| Reserved | 8 |
| } | |
| } | |
| } | |
| for (i=0;i<N1;i++) { | |
| Stuffing_byte | 8 |
| } | |
| for (i=0;i<N2;i++) { | |
| PES_packet_data_byte | 8 |
| } | |
| } | |
| else if (stream_id == program_stream_map stream_id == private_stream_2 stream_id == ECM stream_id == EMM stream_id == program_stream_directory stream_id == DSMCC_stream) stream_id == ITU-T Rec. H.222.1 type E stream { | |
| for (i=0;i<PES_packet_length;i++) { | |
| PES_packet_data_byte | 8 |
| } | |
| } | |
| else if (stream_id == padding_stream) { | |
| for (i=0;i<PES_packet_length;i++) { | |
| Padding_byte | 8 |
| } | |
| } | |
| } | |

2.2.7.2 Definizione semantica dei campi del PES packet

packet_start_code_prefix È un codice di 24 bits. Il suo valore è '0000 0000 0000 0000 0000 0001' (0x000001) ed insieme col stream_id costituisce il codice iniziale del pacchetto (packet_start_code). Permette al decodificatore di riconoscere l'inizio di ogni pacchetto.

stream_id È un campo di 8 bits. Il suo valore identifica inequivocabilmente l' ES che trasporta. A seconda di quale tipo di ES identifichi, può prendere alcuni valori determina oppure altri.

In un Transport Stream, lo stream_id può valere qualsiasi valore che descriva correttamente il tipo di ES secondo la tavola 2.2.4-3. Ad ogni modo è la PSI, attraverso la PMT, qui ha il compito di identificare il tipo di ES.

Tavola 2.2.4-3 Stream_id assignments

| Stream_id | stream coding |
|-------------------------|--|
| 1011 1100 | Program_stream_map |
| 1011 1101 | private_stream_1 (PES packets che trasportano dati privati. Sono presenti soltanto i campi precedenti al PES_header_data_length) |
| 1011 1110 | padding_stream |
| 1011 1111 | private_stream_2(PES packets che trasportano dati privati. Sono presenti soltanto i primi tre campi) |
| 110x xxxx | ISO/IEC 13818-3 or ISO/IEC 11172-3 audio stream number x xxxx |
| 1110 xxxx | ITU-T Rec. H.262 ISO/IEC 13818-2 or ISO/IEC 11172-2 video stream number xxxx |
| 1111 0000 | ECM_stream |
| 1111 0001 | EMM_stream |
| 1111 0010 | ITU-T Rec. H.222.0 ISO/IEC 13818-1 Annex A or ISO/IEC 13818-6_DSMCC_stream |
| 1111 0011 | ISO/IEC_13522_stream |
| 1111 0100 | ITU-T Rec. H.222.1 type A |
| 1111 0101 | ITU-T Rec. H.222.1 type B |
| 1111 0110 | ITU-T Rec. H.222.1 type C |
| 1111 0111 | ITU-T Rec. H.222.1 type D |
| 1111 1000 | ITU-T Rec. H.222.1 type E |
| 1111 1001 | ancillary_stream |
| 1111 1010 ... 1111 1110 | reserved data stream |
| 1111 1111 | program_stream_directory |

PES_packet_length È un campo di 16 bits . Stabilisce il numero di bytes che formano il PESp a partire dall'ultimo byte di questo stesso campo. Un valore di '0' indica che la lunghezza del PES packet non è ne specificata ne limitata, ed è permessa soltanto nei PESps nei quali il payload è formato da bytes di ES video contenuti in TSps.

PES_scrambling_control È un campo di 2 bits. Indica la modalità di criptaggio (scrambling mode) del payload del PESp. Quando il criptaggio viene realizzato a livello PES, lo header dei PESp, inclusi i campi opzionali se sono presenti, non è criptato.

Tavola 2.2.4-4 PES scrambling controllo values

| Valori | Descrizione |
|--------|----------------------|
| 00 | Non criptato |
| 01 | Definito dall'utente |
| 10 | Definito dall'utente |
| 11 | Definito dall'utente |

PES_priority È un campo di 1 bit. Indica la priorità del payload di questo PESp. Un '1' indica una priorità alta del payload rispetto ai payload dei PESps che hanno questo campo a '0'. Un multiplexer può fare uso del PES_priority bit per dare priorità ad alcuni dati all'interno di un ES. Questo campo non può essere modificato dal sistema di trasporto, nel caso si voglia operare in questo contesto si può usare il transport_priority dei TS packets.

data_alignment_indicator È un flag di 1 bit. Indica se l'inizio del payload del pacchetto è allineato (coincide) con l'inizio di qualche unità video o audio (AU o VU). Quando vale '1', lo header del PESp è immediatamente seguito da un codice di inizio video o dalla syncword audio. Se vale '0' non è definito quale è l'allineamento.

Nel caso del video è il descriptor data_stream_alignment_descriptor che indica il tipo di allineamento, però se questo non è presente, per difetto, si tratta dell'inizio di uno slice o di una unità d'accesso.

copyright È un campo di un bit. Quando vale '1' indica che il materiale associato al payload del PESp è protetto da copyright. Quando vale '0' non viene definito se il materiale è protetto da copyright.

Il copyright è associato a tutto l' ES trasportato da questo PES packet, e viene definito attraverso un copyright_descriptor.

originale_or_copy È un campo di un bit. Quando vale '1' il contenuto del payload del PESp associato è originale. Quando vale '0' il contenuto del payload del PESp è una copia.

PTS_DTS_flag È un campo di 2 bits. Quando vale '10' il campo PTS è presente nello header del PESp. Quando vale '11' sono presenti i campi PTS i DTS; quando vale '00' nessuno dei due campi è presente. Il valore '01' non è ammesso.

ESCR_flag È un campo di un bit. Se vale '1' indica che i campi base ed estensione del ESCR sono presenti nello header PESp. Se vale 0 indica che nessun campo ESCR è presente.

ES_rate_flag È un campo di un bit. Quando vale '1' indica la presenza del campo ES_rate nello header PESp. Quando vale '0' indica la non presenza del stesso campo.

DSM_trik_mode_flag È un flag di un bit. Quando vale '1' indica la presenza di un trick mode. Quando vale '0' indica la non presenza del stesso campo.

additional_copy_info_flag È un campo di un bit. Quando vale '1' indica la presenza del campo additional_copy_info. Quando vale '0' indica la non presenza del stesso campo.

PES_CRC_flag È un campo di un bit che. Quando vale '1' indica la presenza del campo CRC nel PESp. Quando vale '0' indica che il CRC non è presente.

PES_extension_flag È un campo di un bit. Quando vale '1' indica l'esistenza del campo di estensione nello header PESp. Quando vale '0' indica che il stesso campo non esiste.

ES_header_data_length È un campo di 8 bits. Stabilisce il numero totale di bytes occupati dai campi opzionali ed i bytes di riempimento contenuti nello header PES. La presenza dei campi opzionali è indicata dai 5 bits che precedono il campo PES_header_data_length.

Maker_bit È un campo di un bit di valore '1'.

PTS (presentation time stamp)

Il PTS è un numero di 33 bits suddiviso in tre parti. Può essere presente soltanto quando il PES packet contiene l'inizio di qualche unità d'accesso. Indica l'istante di presentazione della prima unità d'accesso che comincia in questo PES packet. Il valore del PTS viene specificato in unità del periodo della frequenza del clock del sistema diviso 300 (90 KHz). Il suo valore si deriva dall' equazione seguente:

$$PTS(k) = \left((system_clock_frequency \times tp_n(k)) \text{ DIV } 300 \right) \% 2^{33}$$

dove $tp_n(k)$ è il tempo di presentazione previsto dal codificatore per l'unità di presentazione k dell'ES n . L'ES n è quello trasportato da questo PES packet.

Per le unità di presentazione audio, video in sequenze low_delay e nelle immagini-B, il tempo di presentazione $tp_n(k)$ è uguale al tempo di decodificazione $tdn(k)$ (vedere campo seguente).

In generale, per immagini I e P di sequenze non-low_delay, il tempo di presentazione $tp_n(k)$ è uguale al tempo di decodifica della seguente immagine I o P trasmessa, $tdn(k+1)$.

DTS (decoding time stamp)

Il DTS è un numero di 33 bits diviso in tre parti . Può essere presente soltanto quando il PES packet contiene l'inizio di qualche unità d'accesso. Indica l'istante di decodificazione della prima unità d'accesso che comincia in questo PES packet. Il valore del DTS viene specificato in unità del periodo della frequenza del clock del sistema diviso 300 (90 kHz). Il suo valore si deriva della seguente equazione :

$$DTS(j) = \left((system_clock_frequency \times tdn(j)) \text{ DIV } 300 \right) \% 2^{33}$$

dove $tdn(j)$ è il tempo di decodificazione previsto dal codificatore per la unità d'accesso j dell'ES n . L'ES n è quello trasportato da questo pacchetto.

Può essere presente soltanto quando è presente anche il campo PTS, e quando gli istanti di presentazione e decodificazione differiscono. Ciò avviene nelle immagini I e P di sequenze non-low_delay.

ESCR_base; ESCR_extension(ES Clock Reference) È un campo di 42 bits che ha la stessa struttura del campo PCR. Si utilizza solo quando il pacchetto appartiene a un PES stream e la sua funzione equivale a quella di un PCR in un TS.

Un PES stream è una struttura definita dall'Mpeg-2 Systems che in questo lavoro non abbiamo contemplato. Diciamo soltanto che è simile a un Program Stream ma che invece di utilizzare il campo SCR, e il pogram_multiplex_rate, utilizza l' ESCR ed il ES_rate.

ES_rate (elementary stream rate) È un numero intero senza segno di 22 bits che stabilisce il rate al quale l' STD viaggia e i bytes che formano il PES stream.

Trick mode control È un campo di 3 bits che indica quale trick mode è applicato al ES video trasportato da questo pacchetto. Nel caso di altri tipi di ES, il significato di questo campo e dei seguenti 5 bits è indefinito.

Tavola 2.2.4-5 Trick mode control values

| valori | descrizione |
|-------------|--------------|
| '000' | Fast forward |
| '001' | Slow motion |
| '010' | Freeze frame |
| '011' | Fast reverse |
| '100' | slow reverse |
| '101'-'111' | reserved |

Trick mode è la terminologia che si utilizza per identificare le funzioni di marcia rapida e lenta, avanti e indietro e fermo immagine o pausa. Mpeg-2 contempla queste funzioni per applicazioni di immagazzinamento, come può essere il DVD, e per la televisione interattiva.

Per sequenze progressive (per sequenze interlacciate), quando il flag `trick_mode` è falso, il numero di volte (N) che una immagine è visualizzata viene indicato specificato per ogni immagine attraverso i campi `repeat_first_field` e `top_field_first` (`repeat_first_field` e `progressive_frame`) nel caso dell'Mpeg-2 Video, ed è specificato attraverso la sequenza di intestazione nel caso del Mpeg -1 video.

Quando il flag `trick_mode` è vero, il numero di volte che viene mostrata ogni immagine può variare a seconda del tipo di trick mode.

Fast forward – Quando il campo `trick_mode_control` vale '000' indica: marcia rapida in avanti e comporta la presenza nei 5 bits seguenti dei campi: `intra_slice_refresh`, `field_id` i `frequency_truncation`.

Slow motion – Quando il campo `trick_mode_control` vale '001' indica marcia lenta e comporta la presenza nei 5 bits seguenti dei campi: `intra_slice_refresh`, `field_id` i `frequency_truncation`.

Nel caso di sequenze progressive, l'immagine è mostrata $N * rep_cntrl$ volte (dove N è definito precedentemente).

Nel caso del Mpeg-1 e Mpeg -2 Video, per sequenze progressive l'immagine è visualizzata per $N * rep_cntrl$ volte la durata di una immagine.

Nel caso del Mpeg-2 Video, per sequenze interlacciate, l'immagine è visualizzata per $N * rep_cntrl$ volte la durata di un campo. Il primo campo è visualizzato $(N/2) * rep_cntrl$ volte ed il secondo $N - (N/2) * rep_cntrl$ volte.

Freeze frame – Quando il campo `trick_mode_control` vale '010' indica un fermo-immagine o pausa, e comporta la presenza nei 5 bits seguenti dei campi: `intra_slice_refresh`, `field_id` i `frequency_truncation`.

Il campo `field_id` identifica quale campo/i è visualizzato.

Fast reverse – Quando il campo `trick_mode_control` vale '011' indica marcia rapida all'indietro, e comporta la presenza nei 5 bits seguenti dei campi: `intra_slice_refresh`, `field_id` i `frequency_truncation`.

Slow reverse – Quando il campo `trick_mode_control` vale '100' indica marcia lenta all'indietro e comporta la presenza nei 5 bits seguenti dei campi: `intra_slice_refresh`, `field_id` i `frequency_truncation`.

La durata della presentazione di ogni immagine è la stessa che per il caso Slow motion.

Quando il valore di questo campo cambia o le operazioni trick_mode cessano, qualsiasi delle seguenti combinazioni può verificarsi:

- Discontinuità della base di tempo
- Discontinuità nella decodifica
- Discontinuità del continuity_counter

Field_id È un campo di 2 bits che indica quale campo/i dovrebbe venir visualizzato. Viene codificato secondo la seguente tavola:

Tavola 2.2.4-6 Field_id field

| Valori | Descrizione |
|--------|--------------------------------|
| '00' | Mostrare soltanto top field |
| '01' | Mostrare soltanto bottom field |
| '10' | Mostrare il frame completo |
| '11' | riservato |

Intra_slice_refresh È un campo di un bit. Quando vale '1' indica che possono mancare macroblocchi fra gli slices video dei PES packets. Quando vale '0' ciò non può succedere.

Il decodificatore deve sostituire i macrobloccs che mancano con macroblocchi situati nella stessa posizione delle immagini decodificate precedentemente.

Frequency_truncation È un campo di 2 bits. Indica che un gruppo ristretto di coefficienti può essere stato utilizzato per la codifica dei dati video di questo PES packet. I valori vengono definiti nella seguente tavola:

Tavola 2.2.4-7

| Valori | Descrizione |
|--------|---|
| '00' | Soltanto il coefficiente DC non è zero |
| '01' | Soltanto i 3 primi coefficienti non sono zero |
| '10' | Soltanto i 6 primi coefficienti non sono zero |
| '11' | Tutti i coefficienti possono non essere zero |

Rep_cntrl È un campo di 5 bits. Indica il numero di volte che ogni campo di una immagine interlacciata deve essere visualizzato, o il numero di volte che una immagine progressiva deve essere mostrata. Nel caso di una immagine interlacciata si mostra prima il campo alto (top field) o basso (bottom field) a seconda del valore del campo dello header della sequenza video: top_field_first.

Additional_copy_info È un campo di 7 bits. Contiene dati privati relativi alle informazioni sul copyright.

Previous_PES_packet_CRC È un campo di 16 bits. Contiene il CRC calcolato a partire dal polinomio: $x^{16}+x^{12}+x^5+1$

Si calcola soltanto per i bytes di dati perchè i dati delle intestazioni dei PESp possono essere modificate durante il trasporto.

È stato pensato per essere usato in funzioni di manutenzione del network di comunicazione (per localizzare fonti di errore). Non è stato pensato per l'utilizzo nei decodificatori di ES.

PES_private_data_flag È un campo di un bit. Quando vale '1' indica che lo header PES packet contiene dati privati. Quando vale '0' indica che non ci sono dati privati nello header PES.

Pack_header_field_flag È un campo di un bit. Quando vale 1 indica che un Mpeg-1 pack header o una pack header Program Stream è contenuto in questa intestazione (header) di PES packet. In un TS, quando vale '0' indica la non presenza di nessun pack header nell'intestazione PES.

La sua funzione è vincolata ai TS che trasportano dati di un Program Stream.

Program_packet_sequence_counter_flag È un campo di un bit. Quando vale '1' indica che i campi program_packet_sequence_counter, MPEG1_MPEG2_identifier e original_stuff_length sono presenti in questo PESp. Quando vale '0' indica la non presenza di questi campi nell'intestazione PES.

PES_extension_flag_2 È un campo di un bit che se vale '1' indica la presenza del campo PES_extension_field_length e dei campi associati. Se vale '0' indica che il campo PES_extension_field_length ed i campi associati non sono presenti.

PES_private_data È un campo di 16 bytes. Contiene dati privati. Questi due bytes combinati con i bytes precedenti e successivi, non possono essere uguali al packet_start_code_prefix (0x000001).

Pack_field_length È un campo di 8 bits che indica la lunghezza, in bytes, del campo pack_header(). Il campo pack_header() è quello dove si trasportano le intestazioni pack_header dei Program Streams, o i Mpeg-1 System Streams, quando questi sono trasportati in Transport Streams (TS).

Program_packet_sequence_counter È un campo di 7 bits. È un contatore opzionale che viene incrementato ad ogni PESp successivo di un PS o di un ISO/IEC 11172-1 Stream o dei PESps associati alla definizione di un solo programma in un TS, e fornisce una funzionalità simile a quella del contatore di continuità. Ciò permette a una applicazione di recuperare la sequenza di PESp di un PS o la sequenza originale del ISO/IEC 11172-1 Stream originale. Il contatore viene resettato al raggiungimento del suo valore massimo. Non possono verificarsi ripetizioni di PESp. Conseguentemente, due PESp consecutivi nel programma multiplexato non devono avere valori del program_packet_sequence_counter identici.

MPEG1_MPEG2_identifier È un flag di un bit che quando vale '1' indica che questo PESp trasporta informazioni di uno stream Mpeg-1. Quando vale '0' indica che questo PESp trasporta informazioni di un PS.

PES_extension_field_length È un campo di 7 bits che stabilisce la lunghezza, in bytes, dei dati che seguono questo campo all'interno del campo di estensione PES, può contenere anche byte riservati (?).

Stuffing_byte (byte di riempimento) È un campo di 8 bits di valore fisso '1111 1111'. Può essere inserito dall'encoder (codificatore); per esempio per soddisfare i requisiti del canale. È un byte che viene scartato dal decoder (decodificatore). Non ci possono essere più di 32 bytes di riempimento in una intestazione PESp.

PES_packet_data_byte questo campo è formato da bytes continui e nello stesso ordine di dati del ES indicato dallo stream_id o dal PID. Quando l'ES è un Mpeg-2 Video o Audio, i suoi bytes vengono allineati con i bytes del PES packet. Il numero di bytes (N) dell'ES contenuti nel PES packet è specificato dal campo PES_packet_length. Il valore N è uguale a quello indicato dal campo PES_packet_length meno il numero di bytes che ci sono fra l'ultimo byte del campo PES_packet_length ed il primo del PES_packet_data_byte.

Nel caso di private_stream_1, private_stream_2, ECM_stream, o EMM_stream, il contenuto del PES_packet_data_byte è definibile dal distributore.

Padding_byte È un campo di 8 bits di valore '1111 1111'. Viene scartato dal decoder (decodificatore).

2.2.7.3. Transport Stream Packet

Nei TS packets, la funzione dei campi non è più così vincolata al livello di compressione. I TSp sono diretti a facilitare il trasporto e la rigenerazione dei PES packets nel ricevitore.

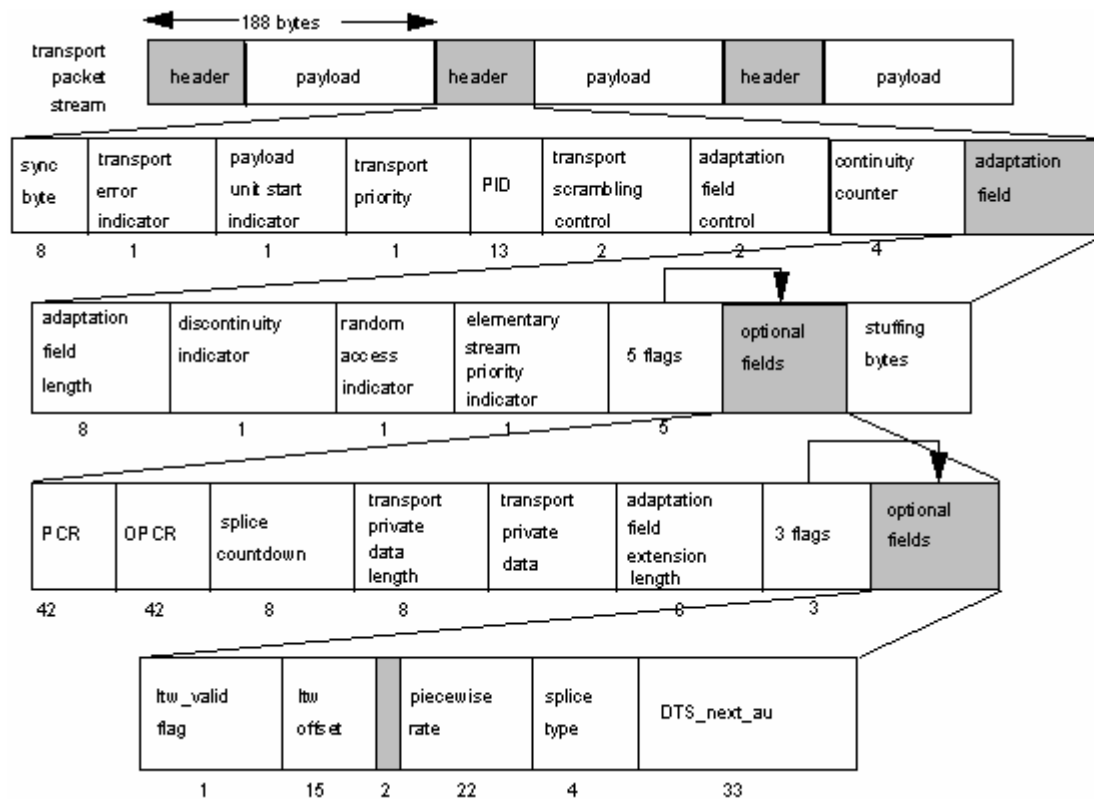


Immagine 2.2.4-2

Tavola 2.2.4-8 Transport Stream Packet

| Sintassi | Numero di bits |
|--|----------------|
| transport_packet(){ | |
| sync_byte | 8 |
| transport_error_indicator | 1 |
| payload_unit_start_indicator | 1 |
| transport_priority | 1 |
| PID | 13 |
| transport_scrambling_control | 2 |
| adaptation_field_control | 2 |
| continuity_counter | 4 |
| if(adaptation_field_control=='10' adaptation_field_control=='11'){ | |
| Adaptation_field() | |
| } | |
| if(adaptation_field_control=='01' adaptation_field_control=='11') { | |
| For (i=0;i<N;i++){ | |
| data_byte | 8 |
| } | |
| } | |
| } | |

2.2.7.4 Definizione semantica dei campi di un Transport Stream packet

sync_byte È il byte di sincronismo. Ha sempre lo stesso valore '01000111' (0x47). Questo valore non si può ripetere in altri campi del TS packet. Il fatto che ogni 188 bytes si ripeta il sync_byte, facilita ai decoders l'identificazione dell'inizio dei pacchetti.

transport_error_indicator È un flag di un bit. Quando ha il valore '1', indica che come minimo esiste un bit erroneo incorreggibile nel TSp associato. Questo bit può essere posto a '1' per entità esterne al livello di trasporto. Quando il suo valore è uguale a '1', non può essere posto a '0' a meno che i bits erronei siano stati corretti.

payload_unit_start_indicator È un flag di un bit che ha significato solo per i TSps che trasportano PESps o dati PSI.

Quando il payload del TSp contiene dati di un PESp, l'indicatore ha il seguente significato:
 un '1' indica che il payload di questo TSp comincerà con il primo byte di un PESp
 uno '0' indica che nessun PESp comincia all'interno di questo TSp.
 Si l'indicatore vale '1', soltanto un PESp può iniziare in questo TSp.

Quando il payload del TSp contiene dati PSI, l'indicatore ha il seguente significato:
 un '1' indica che il TSp trasporta il primo byte di una sezione PSI, e che il primo byte del payload di questo TSp è il pointer_field. Le sezioni PSI non devono coincidere con l'inizio del payload dei TSps, perciò il campo pointer_field ha il compito di indicare dove si trova l'inizio della sezione.
 uno '0' indica che il TSp non trasporta il primo byte di una sezione PSI e pertanto il campo pointer_field non è presente.

Per pacchetti vuoti l'indicatore è '0'.

transport_priority È un indicatore di un bit. Quando questo vale '1' indica che il pacchetto associato è prioritario rispetto al resto dei pacchetti che hanno lo stesso PID però che hanno l'indicatore a '0'. Il meccanismo di trasporto può usare questo indicatore per dare la priorità a dati all'interno di un ES. A seconda dell'applicazione il campo transport_priority può essere utilizzato senza tener conto del valore PID o soltanto per un solo PID. Questo campo può essere modificato da encoders o decoders specifici di canale.

PID (Packet Identifier) È un campo di 13 bits. Si utilizza per distinguere i TS packets che trasportano dati di un ES da quelli che trasportano dati di altri ESs. Dei 2^{13} valori possibili, 17 sono riservati (come si può vedere nella tavola 2.2.4-9). Ciò significa che vi sono 8175 valori che possono essere assegnati a differenti ES, e rappresenta il massimo numero di ESs che possono esserci in un sol Transport Stream. È responsabilità del distributore, assicurare che ogni ES abbia assegnato un solo PID. L'Mpeg non definisce quale valore deve ricevere ogni ES, ciò rimane a discrezione del distributore.

Soltanto i transport packets con PID di valori 0x0000, 0x0001, e 0x0010-0x1FFE possono trasportare il campo PCR.

Tavola 2.2.4-9 PID

| Valore | Descrizione |
|--------------------------|--|
| 0x0000 | Program Association Table |
| 0x0001 | Conditional Access Table |
| 0x0002-0x000F | Riservato |
| 0x00010 ... 0x1FFE | Può essere assegnato al network_PID, Program_map_PID, elementary_PID, o per altri propositi. |
| 0x1FFF | Pacchetti vuoti |

Transport_scrambling_control Questo campo di 2 bits indica la modalità di criptaggio (scrambling mode) del payload del TSp. L'intestazione del TSp e l'Adaptation field (quando è presente) non possono essere criptati poiché contengono parametri che possono essere letti e modificati durante il suo trasporto. Nei pacchetti senza payload, il transport_scrambling_control vale '00'.

Tavola 2.2.4-10 Valori Scrambling controllo

| Valore | Descrizione |
|--------|----------------------|
| 00 | Non criptato |
| 01 | Definito dall'utente |
| 10 | Definito dall'utente |
| 11 | Definito dall'utente |

adaptation_field_control Questo campo di 2 bits indica se lo header del TSp è seguito dall'adaptation field e/o del payload.

Tavola 2.2.4-11 Valori Adaptation field

| Valore | Descrizione |
|--------|--|
| 00 | Riservato per usi futuri da ISO/IEC |
| 01 | nessun adaptation_field, soltanto payload |
| 10 | soltanto adaptation_field , soltanto payload |
| 11 | adaptation_field seguito da payload |

continuity_counter È un campo di 4 bits. Concettualmente questo campo prende il valore di un contatore che è associato al PID del TSp. Questo contatore incrementa di una unità ogni volta che viene inviato un TSp su questo PID. Pertanto, ogni TSp ha un valore di continuity_counter incrementato di una unità rispetto al precedente TSp di uguale PID.

Il contatore si azzerà ogni volta che arriva al suo massimo (15), e non si incrementa quando il TSp non contiene payload (adaptation_field_control vale '00' o '10').

I pacchetti duplicati hanno lo stesso valore di continuity_counter che il pacchetto originale.

In un Transport Stream, si definiscono i pacchetti duplicati come quelli in cui ogni byte è una copia del pacchetto originale con l'eccezione del campo PCR, che se è presente, ha il valore che gli corrisponde. Devono sempre contenere payload, e possono essere inviati soltanto due TSps consecutivi sullo stesso PID.

Il continuity_counter si considera continuo quando questo differisce di una unità positiva rispetto al contatore del TSp precedente dello stesso PID, o quando ci siano le condizioni di non incremento menzionate precedentemente (adaptation_field_control uguale a '00' o '10', o pacchetti duplicati).

Il continuity_counter può essere discontinuo quando il valore del discontinuity_indicator vale '1' (vedere 2.2.7.5). Nel caso di pacchetti senza payload il valore del continuity_counter non è definito.

data_byte I bytes di dati devono essere bytes di dati continui (nello stesso ordine) appartenenti ai PESps, alle sezioni PSI, a bytes di riempimento posteriori alle sezioni PSI, o a dati privati all'interno di queste strutture come indicato dal proprio il suo PID. Nel caso di pacchetti senza payload (PID uguale a '0x1FFF'), i bytes di dati possono essere qualsiasi. Il numero di bytes di dati è stabilito in 184 meno il numero di bytes dell'adaptation field.

2.2.7.5. Campo d'adattamento (Adaptation field)

Tavola 2.2.4-12 Transport Stream adaptation field

| Sintassi | Numero di bits |
|--|----------------|
| Adaptation_field() { | |
| Adaptation_field_length | 8 |
| if(adaptation_field_length >0) { | |
| Discontinuity_indicator | 1 |
| Random_access_indicator | 1 |
| Elementary_stream_priority_indicator | 1 |
| PCR_flag | 1 |
| OPCR_flag | 1 |
| Splicing_point_flag | 1 |
| Transport_private_data_flag | 1 |
| Adaptation_field_extension_flag | 1 |
| if(PCR_flag == '1') { | |
| program_clock_reference_base | 33 |
| Reserved | 6 |
| program_clock_reference_extension | 9 |
| } | |
| if(OPCR_flag == '1') { | |
| originale_program_clock_reference_base | 33 |
| Reserved | 6 |
| originale_program_clock_reference_extension | 9 |
| } | |
| if (splicing_point_flag == '1') { | |
| splice_countdown | 8 |
| } | |
| if(transport_private_data_flag == '1') { | |
| transport_private_data_length | 8 |
| for (i=0; i<transport_private_data_length;i++){ | |
| private_data_byte | 8 |
| } | |
| } | |
| if (adaptation_field_extension_flag == '1') { | |
| adaptation_field_extension_length | 8 |
| ltw_flag | 1 |
| piecewise_rate_flag | 1 |
| seamless_splice_flag | 1 |
| reserved | 5 |
| if (ltw_flag == '1') { | |
| ltw_valid_flag | 1 |
| ltw_offset | 15 |
| } | |

| | |
|-----------------------------------|-----------|
| if (piecewise_rate_flag == '1') { | |
| reserved | 2 |
| piecewise_rate | 22 |
| } | |
| if (seamless_splice_flag == '1'){ | |
| splice_type | 4 |
| DTS_next_AU[32..30] | 3 |
| marker_bit | 1 |
| DTS_next_AU[29..15] | 15 |
| marker_bit | 1 |
| DTS_next_AU[14..0] | 15 |
| marker_bit | 1 |
| } | |
| for (i=0;i<N;i++) { | |
| reserved | 8 |
| } | |
| } | |
| For (i=0;i<N;i++){ | |
| stuffing_byte | 8 |
| } | |
| } | |
| } | |

2.2.7.6 Definizione semantica dei campi dell'Adaptation field

adaptation field length È un campo di 8 bits. Stabilisce il numero di bytes che forma l'adaptation field che seguono immediatamente l'adaptation_field_length. Se il suo valore è '0' si sta inserendo un solo byte di riempimento nel TSp. Quando è presente anche il payload, il valore dell'adaptation_field_length è compreso fra 0 i 182, e se non è presente il suo valore è 183.

Per i TSps che trasportano PESps, è necessario un riempimento quando i dati di un PESp non sono sufficienti per riempire completamente i bytes di payload del TSp. Il riempimento si ottiene attraverso la definizione di un adaptation field più lungo della somma delle lunghezze degli elementi di dati che li formano, in modo che i bytes di payload rimanenti dietro al campo d'adattamento, si incastrino perfettamente nei dati disponibili del PESp. Lo spazio extra del adaptation field si riempie con i bytes di riempimento.

Questo è l'unico metodo di riempimento ammesso nei TSps che trasportano PESps. Per i TSps che trasportano informazione PSI esiste un metodo alternativo.

discontinuity_indicator È un campo di un bit .Quando vale '1' indica che il flag di discontinuità è 'vero' per il TSp associato. Quando l'indicatore vale '0' o non è presente, il flag di discontinuità è 'falso'. Può indicare due tipi di discontinuità, discontinuità della base di tempo del sistema e discontinuità del contatore di continuità.

Una discontinuità della base di tempo del sistema è indicata attraverso l'utilizzo dell'indicatore di discontinuità del TSp di PID uguale a quello designato come a PCR_PID. Una discontinuità della base di tempo si verifica quando il PCR contenuto in un TSp di PID=PCR_PID, rappresenta un campione di un nuovo clock di sistema (STC) per il programma associato. In questo stesso pacchetto il discontinuity_indicator vale '1'.

Il discontinuity_indicator può anche valere '1' nei TSps (di PID=PCR_PID) che precedono il pacchetto che contiene il primo PCR della nuova base di tempo. In questo caso, l'indicatore di discontinuità

continua con lo stesso valore in tutti i TSpS con PID=PCR_PID fino al TSp che contiene il primo PCR della nuova base di tempo.

Dopo una discontinuità nella base di tempo, si devono ricevere come minimo due PCR's della nuova base di tempo, prima che possa verificarsi una nuova discontinuità.

Prima che si verifichi una discontinuità della base di tempo del sistema, non può giungere al decodificatore nessun TSp che contenga un PTS o DTS della nuova base di tempo. E neanche può giungere un TSp che contenga un PTS o DTS della base di tempo precedente, dopo che si sia prodotta una discontinuità della base di tempo del sistema.

Una discontinuità del continuity_counter è indicata anche, in qualsivoglia TSp, attraverso l'utilizzo del discontinuity_indicator. Quando l'indicatore di discontinuità vale 1 in qualsiasi TSp con PID differente a quello designato come PCR_PID, il continuity_counter di questo pacchetto può essere discontinuo rispetto il TSp precedente con lo stesso PID. Nei TSpS di PID uguale al PCR_PID, il continuity_counter può essere discontinuo soltanto nei pacchetti in cui ha luogo anche in essi una discontinuità della base di tempo. Una discontinuità del continuity_counter si produce quando in uno stesso TSp, l'indicatore di discontinuità è 'vero' e il continuity_counter è discontinuo rispetto il TSp precedente con uguale PID. I due pacchetti precedenti e con PID uguale al pacchetto nel quale avviene la discontinuità, possono avere il continuity_indicator a '1'. Dal momento in cui l'indicatore di continuità acquisisce il valore '1' fin a che questo ritorna a valere '0' può avere luogo soltanto una discontinuità del continuity_counter.

Dopo una discontinuità del continuity_counter per un TSp che contiene dati di un determinato ES, il primo byte dello stesso ES ricevuto e contenuto in un TSp d'uguale PID al precedente, è il primo byte di un punto d'accesso di un ES. Nel caso di video può essere anche una sequence_end_code seguita da un punto d'accesso. Definiamo ora il punto d'accesso di un ES:

- Video – Il primo byte dello header di una sequenza de video.
- Audio – Il primo byte di un frame audio.

Dopo il passaggio di un valore '1' nel discontinuity_indicator di un TSp che contiene informazioni PSI, può verificarsi una discontinuità soltanto nei version_number delle sezioni PSI. Quando ha luogo questa discontinuità, una versione del TS_program_map_sections del programma in questione deve essere inviata con una section_length=13 ed un current_next_indicator=1, in modo tale che non ci siano program_descriptors né elementary streams descritti. Ciò deve essere seguito da una versione del TS_program_map_section per ogni programma chiamato in causa, in modo che contenga una definizione completa del programma, il version_number incrementato di una unità ed il current_next_indicator=1. Ciò indicherà un cambio di versione nei dati PSI.

random_access_indicator È un campo di un bit. Indica che il TSp attuale, e possibilmente anche i seguenti TSp's con lo stesso PID, contengono qualche informazione per permettere l'accesso aleatorio in questo punto. Concretamente, quando il campo vale '1', il seguente PESp che inizi nel payload qualche TSp con lo stesso PID, contiene il primo byte dello header di una sequenza video o il primo byte di un frame audio. Per di più, nel caso del video, il PESp che contiene la prima immagine successiva allo header (intestazione) della sequenza contiene un PTS (Program TimeStamp). E nel caso dell'audio, il PESp che contiene il primo byte del frame d'audio contiene un PTS.

Nei TS packets del PCR_PID, il random_access_indicator può valere '1' soltanto nei pacchetti che contengono i campi PCR.

elementary_stream_priority_indicator È un campo di un bit. Indica, fra i TSpS con lo stesso PID, la priorità dei dati del ES trasportati nel payload. Un '1' indica che il payload ha una priorità più alta che il payload degli altri TSpS. Nel caso del video, questo campo può valere '1' soltanto se il payload contiene uno o più bytes di uno slice di una immagine I. Uno '0' indica che il payload ha la stessa priorità di tutti gli altri pacchetti che non hanno questo campo a '1'.

PCR_flag Il PCR_flag è un flag di un bit. Un valore di '1' indica che l'adaptation field contiene i due campi PCR. Un valore di '0' indica che l'adaptation field non contiene nessun campo PCR.

OPCR_flag È un flag di un bit. Quando vale '1' indica che l'adaptation field contiene i due campi OPCR. Quando vale '0' indica che l'adaptation field non contiene nessun campo OPCR.

splicing_point_flag È un flag di un bit. Quando vale '1' indica la presenza del campo splice_countdown nell'adaptation field associato e l'esistenza di un punto di splicing. Quando vale '0' indica che questi campi non sono presenti.

transport_private_data_flag È un flag di un bit. Se vale '1' indica che l'adaptation field contiene uno o più bytes di dati privati. Se vale '0' indica che l'adaptation field non contiene dati privati.

adaptation_field_extension_flag È un flag di un bit. Se vale '1' indica la presenza dell'estensione dell'adaptation field. Quando vale '0' indica che l'estensione dell'adaptation field non è presente.

program_clock_reference_base; program_clock_reference_extension Il program_clock_reference (PCR) è un campo di 42 bits codificato in due parti. La prima parte, il program_clock_reference_base, è un campo di 33 bits il valore del quale viene dato dal PCR_base(i), (equazione 2.2.3-1). La seconda parte, program_clock_reference_extension, è un campo di 9 bits il valore del quale viene dato dal PCR_ext(i), (equazione 2.2.3-2). Il campo PCR indica il tempo d'arrivo previsto del byte che contiene l'ultimo bit del program_clock_reference_base a all'ingresso del T-STD.

original_program_clock_reference_base; original_program_clock_reference_extension L'original_program_clock_reference (OPCR) è un campo opzionale di 42 bits che ha la stessa sintassi del campo PCR. La presenza del OPCR è indicata tramite l'OPCR_flag. Il campo OPCR può essere inserito soltanto nei TSps che contengano il campo PCR. Gli OPCR's sono ammessi sia nei TS di uno o più programmi.

Il campo OPCR partecipa nella ricostruzione di un TS di un solo programma (single program TS) a partire da un altro TS. Quando si ricostruisce il single program TS originale, l'OPCR può essere copiato nel campo PCR. Il valore risultante del PCR è valido solo se il single program TS originale è ricostruito esattamente nella sua totalità. Questo include come minimo tutti i pacchetti di dati PSI e 'private' che erano presenti nel TS originale, e possibilmente includerebbe anche altre strutture private. Significa anche che l'OPCR è una copia identica di questo PCR associato al single program TS originale (??? – qui ci sono degli errori nel testo originale in catalano. *n.d.t.*)

$$OPCR(i) = OPCR_base(i) \times 300 + OPCR_ext(i)$$

dove

$$OPCR_base(i) = \left(\left(system_clock_frequency \times t(i) \right) DIV 300 \right) \% 2^{33}$$

$$OPCR_ext(i) = \left(\left(system_clock_frequency \times t(i) \right) DIV 1 \right) \% 300$$

Il campo OPCR è utilizzato dal re-multiplexer, ma è ignorato dal decodificatore (decoder). Il campo OPCR non può essere modificato né dal multiplexer né dal decodificatore.

splice_countdown È un campo di 8 bits che può rappresentare valori positivi o negativi.

Se si tratta di un valore positivo stabilisce il numero di TSp, con lo stesso PID, che rimangono dopo il TSp associato prima che arrivi un punto di splicing. I TSp duplicati e quelli che contengono soltanto l'adaptation field non sono tenuti in conto. Il punto di splicing è localizzato immediatamente dopo l'ultimo byte del TSp nel quale lo splice_countdown associato arriva a zero. Nel TSp in cui lo splice_countdown arriva a zero, l'ultimo byte codificato nel payload del TSp è l'ultimo byte di un frame audio codificato o di una immagine codificata. Per ciò che riguarda i pacchetti video, la corrispondente unità di accesso può essere finalizzata da una sequence_end_code o può anche non esserlo. I successivi TSps con lo stesso PID possono contenere dati di un ES differente però dello stesso tipo.

Il payload del seguente TSp con lo stesso PID (salvo i pacchetti duplicati o senza payload) cominciano con il primo byte di un PESp. Per l'audio, il payload del PESp comincia con un punto d'accesso. Nel caso del video, il payload del PESp comincia con un punto d'accesso o con una sequence_end_code, seguita da un punto d'accesso. Pertanto bisognerà che l'immagine o il frame audio precedenti si allineino col limite del pacchetto, e se occorre siano riempiti per ottenere questo allineamento. I pacchetti successivi a un punto di splicing possono contenere il campo countdown. In questi casi è un numero negativo di

valore meno n ($-n$), per indicare che il TSp associato è l' n -esimo pacchetto dopo il punto di splicing (i pacchetti duplicati o senza payload non si contano).

transport_private_data_length È un campo di 8 bits. Stabilisce il numero di bytes di dati privati che seguono. Non può indicare che i dati privati eccedano il limite dell'*adaptation field*.

private_data_byte È un campo di 8 bits. Può avere qualsiasi valore che il distributore desideri assegnargli.

adaptation_field_extension_length È un campo di 8 bits. Indica il numero di bytes di dati dell'estensione dell'*adaptation field* che segue, inclusi i bytes riservati se presenti.

ltw_flag (legal time window flag) È un campo di un bit. Quando vale '1' indica la presenza del campo *ltw_offset*.

piecewise_rate_flag È un campo di un bit. Quando vale '1' indica la presenza del campo *piecewise_rate*.

seamless_splice_flag È un campo di un bit. Quando vale '1' indica la presenza dei campi *splice_type* e *DTS_next_AU*. Quando vale '0' indica che nessuno di questi campi è presente.

Questo campo non può valere '1' nei TSps nei quali lo *splice_point_flag* non vale '1'. Quando vale '1' in un TSp nel quale lo *splice_countdown* è positivo, vale '1' anche in tutti i TSps seguenti dello stesso PID e con lo *splice_point_flag* a '1', fino al pacchetto nel quale lo *splice_countdown* arriva a zero (questo pacchetto incluso). Vedere *splice_type*.

ltw_valid_flag (legal time window valid flag) È un campo di un bit. Quando vale 1 indica che il valore del *ltw_offset* è valido. Un valore di '0' indica che il valore del campo *ltw_offset* è indefinito.

ltw_offset Questo è un campo di 15 bits. Il suo valore è definito soltanto se il flag *ltw_valid* vale '1'. Il Legal Time Window è espresso in unità di $1/300$ la frequenza del clock del programma al quale appartiene il PID associato.

La Legal Time Window è un intervallo di tempo associato a un TS packet. Questo assicura che se il pacchetto è mandato al T-SDT all'interno di questo intervallo, non avvenga nessuna violazione nei buffers del decoder.

Il limite superiore di questa finestra di tempo è $t_1(i)$, che si determina a seconda che questo TSp (ancora errori nel testo originale – *n.d.t.*)

offset = $t_1(i) - t(i)$ *ltw_offset* = valore intero di (offset)

dove i è il indice del primo byte di questo TSp, $t(i)$ è l'istante d'arrivo del byte al T-STD, l'offset è il valore contenuto in questo campo.

Nella formula precedente si determina solo il valore di $t_1(i)$ per i pacchetti che contengono il campo *ltw_offset*, ci si riferisca al *piecewise_rate* per sapere come si calcola il resto.

Il limite inferiore, $t_0(i)$, si può determinare a seconda di fattori come la grandezza del buffer MBn e la velocità di trasferimento dei dati fra l' MBn e l' EBn.

L'informazione di questo campo è pensata per dispositivi come remultiplexers che possono aver bisogno di questa informazione per ricostruire lo stato dei buffers MBn.

piecewise_rate È un campo di 22 bits. Viene definito quando il *ltw_flag* ed il *ltw_valid_flag* valgono '1'.

Se viene definito, questo campo è un numero intero positivo che stabilisce un bitrate ipotetico, R , che si usa per definire gli istanti a cui termina la Legal Time Window dei TSps (successivi a questo pacchetto e di uguale PID) che non includono il campo *legal_time_window_offset*.

Assumendo che il primo byte di questo TSp e dei successivi N TSps con lo stesso PID hanno gli indici A_i , A_{i+1} , ..., A_{i+N} , rispettivamente, e che gli ultimi N pacchetti non contengono il campo *legal_time_window_offset*, allora i $t_1(A_{i+j})$ sono determinati da:

$$t_1(A_{i+j}) = t_1(A_i) + j * 188 * 8 \text{ bits/byte} / R \text{ dove } j \text{ va da } 1 \text{ fino a } N$$

Tutti i pacchetti contenuti fra questo pacchetto e il seguente con lo stesso PID che include il campo `legal_time_window_offset` devono essere trattati come se contenessero il valore:

$offset = t1(A_i) - t(A_j)$

dove $t1(.)$ si calcola secondo la formula precedente e dove $t(j)$ è l'istante d'arrivo del byte j al T-STD. Il significato di questo campo non è definito quando è presente in un TSp che non contiene il campo `legal_time_window_offset`.

splice_type Questo è un campo di 4 bits. Indica quali caratteristiche ha il punto di splicing o in quali circostanze deve avere luogo.

Ogni valore dello `splice_type` indica un valore di `splice_decoding_delay` e uno di `max_splice_rate`, secondo le tavole da 2-7 a 2-16 dello Standard Mpeg-2 Systems.

Nel caso dell'audio questi valori sono sempre uguali (`splice_type='0000'`) ma nel caso del video dipendono dal profilo e dal livello.

Lo `splice_decoding_delay` indica la durata della presentazione dell'immagine che termina nel TSp nel quale lo `splice_countdown` arriva a zero, e il tempo che l'ultimo byte della immagine deve rimanere nel buffer EB.

Il buffer EB deve avere la capacità sufficiente per non andare in overflow, se l'entrata si connette ad un punto di splicing, di uno stream con rate uguale al `max_splice_rate`, durante un intervallo di tempo uguale allo `splice_decoding_delay`.

DTS_next_AU (decoding time stamp next unit) Questo è un campo di 33 bits formato da tre parti. Nel caso di decodifiche continue e periodiche attraverso un punto di splicing, questo campo indica il tempo di decodifica della prima unità d'accesso dopo il punto di splicing. Questo tempo di decodifica è espresso nella base di tempo valida per l' TSp nel quale lo `splice_countdown` giunge a zero. A partire dal pacchetto nel quale è contenuto per la prima volta il `DTS_next_AU` per un certo PID, questo campo mantiene lo stesso valore nei seguenti TSps (Transport Stream Packets) con lo stesso PID, fino al pacchetto nel quale lo `splice_countdown` arriva a zero (quest'ultimo pacchetto incluso).

stuffing_byte È un campo di 8 bits. Il suo valore è fisso '1111 1111', ed è inserito dall'encoder per necessità di riempimento. Il decoder lo scarnerà.

2.2.7.7. Transport Streams con più di un programma e con rate variabile.

Come abbiamo già visto, un Transport Stream può contenere più di un programma e può essere trasportato ad un rate variabile. Questi due possibilità però non sono compatibili se il TS trasporta programmi che fanno riferimento a basi di tempo differenti.

Ogni programma trasportato richiede una serie continua di PCR's (Program Clock References) per poter rigenerare l' STC (System Time Clock) nel decoder. Se due programmi hanno basi di tempo differenti, ognuno ha associato un PID_PCR diverso pertanto i campi PCR per questi due programmi sono situati in punti differenti del TS, non sono co-localizzati.

Supponiamo che questi due programmi siano trasportati in un TS con un rate variabile. Si teniamo conto che la differenza fra il tempo d'arrivo di campi PCR dipende dal transport rate e che questo tempo deve corrispondere all'intervallo di tempo esistente fra i due valori PCR's, possiamo concludere che il transport rate può variare soltanto in accordo coi PCR's di un solo programma.

Perché il sistema funzioni correttamente, anche quando i PCR's (e per conseguenza anche i punti del TS dove il rate varia) non sono co-localizzati, il rate con cui il TS dovrebbe giungere al decoder, dovrebbe variare a seconda del programma che si desidera decodificare. Evidentemente ciò non è possibile, e pertanto nemmeno è possibile costruire una lista coerente tutto di ciò che viene inviato per tutto un TS quando questo contiene più di un programma con basi di tempo indipendenti e con rate variabili.

Comunque sia è invece semplice costruire un TS con programmi multipli con basi di tempo indipendenti e rate costante.

2.2.7.8. Trasporto di Program Streams e di Mpeg-1 Systems Streams all'interno di un TS

L' Mpeg-2 Systems contempla la possibilità di trasportare Program Streams, Mpeg-1 System Streams e altri Transport Streams, attraverso la struttura del Transport Stream.

Il Transport Stream è progettato in modo che è possibile realizzare con facilità le seguenti operazioni:

- Estrarre da un Transport Stream i pacchetti corrispondenti a un programma e produrre un nuovo TS con soltanto questo programma.
- Estrarre da uno o più Transport Streams i pacchetti di uno o più programmi, e produrre a partire da questi un nuovo TS.
- Estrarre da un Transport Stream i contenuti di un programma e produrre un nuovo PS.
- Prendere un PS e convertirlo in un Transport Stream per poterlo trasportare in un ambiente con perdite. E una volta trasportato, ricostruire un PS valido, che in certi casi potrà essere identico all'originale.

Per questa ragione i PES packets ed i TS packets dispongono di campi opzionali per supportare questi processi e permettere una ricostruzione semplice dei flussi (streams) rispettivi nel decoder.

Il processo per codificare un PS all'interno di un TS è il seguente:

- I PES packets che formano il PS (PS PES packets) e che contengono dati Mpeg-2 video o audio, Mpeg-1 video o audio, o private_stream_1, sono semplicemente trasportati in TS packets.

Quando si sta ricostruendo il PS nel decodificatore, i dati che contengono questi PES packets sono semplicemente copiati sequenzialmente nel PS che si sta ricostruendo.

- Per i PS PESp che contengono dati di `program_stream_map`, `padding_stream`, `private_stream_2`, `ECM`, `EMM`, `DSM_CC_stream`, o `program_stream_directory`, il processo è differente. Tutti i bytes del PS PESp, salvo il campo `packet_start_code_prefix`, sono copiati nel payload di un nuovo PESp. Come si può vedere alla tavola 2.2.4-2, in questo caso un PES packet soltanto presenta i campi `packet_start_code_prefix`, `stream_id` e `PES_packet_length`, payload a parte. Lo `stream_id` prende sempre il valore corrispondente a un `ancillary_stream` (vedere tavola 2.2.4-3). Questo nuovo PES packet in seguito viene trasportato in TSps.

Quando si sta ricostruendo il PS nel decodificatore, per i PESps con `stream_id` di valore `ancillary_stream_id`, prima si scrive il `packet_start_code_prefix` e poi vengono copiati i bytes del payload.

- Il campo `program_packets_sequence_counter` viene incluso nello header di ogni PESp che trasporta dati di un PS. Ciò permette riprodurre nel decoder l'ordine originale dei PES packets.
- Infine, i `packet_header()`'s di un PS sono trasportati nel TS, nello header del PES packet immediatamente successivo a esso. Concretamente dopo il campo `pack_field_length`.

Il processo per codificare un Mpeg-1 System Stream all'interno di un TS è il seguente:

- Gli streams Mpeg-1 sono trasportati nel TS attraverso la sostituzione degli headers dei pacchetti Mpeg-1 con gli headers dei pacchetti PES. I valori dei campi degli headers dei pacchetti Mpeg-1 vengono copiati nei campi equivalenti degli headers PES.
- Il campo `program_packets_sequence_counter` viene incluso nella nello header di ogni pacchetto PES che trasporta dati di un Mpeg-1 System stream. Ciò permette riprodurre nel decodificatore l'ordine originale dei pacchetti del Mpeg-1 System stream.

Infine, i `packet_header()`'s di un Mpeg-1 System stream sono trasportati nel TS, nello stesso modo con cui vengono trasportati i `packet_header()`'s di un PS.

2.3. Program Specific Information (PSI)

2.3.1. Introduzione alla Program Specific Information

L'Mpeg-2 Systems stabilisce un metodo per la descrizione dei contenuti dei TSps con l'obiettivo di semplificare e guidare i processi di demultiplexaggio e presentazione dei programmi. Questo metodo si concretizza in una specificazione sintattica chiamata Program Specific Information (PSI).

Dato un TS, tutti i TS packets che appartengono ad un ES hanno uno stesso PID, i pacchetti di un altro ES avranno un PID differente. Il demultiplexer può facilmente selezionare un determinato ES semplicemente accettando soltanto i pacchetti col PID corrispondente e può selezionare tutto un programma si conosce quali sono i PIDs del video, audio e dati che lo compongono. Però affinché il decodificatore sappia quali PIDs formano ogni programma è imprescindibile l'utilizzo della Program Specific Information.

La PSI non è altro che un insieme di dati strutturati in quattro tavole:

- Program Association Table (PAT)
- Transport Stream Program Map Table (PMT)
- Network Information Table (NIT)
- Conditional Acces Table (CAT)

Le Tavole PSI

Program Association Table

La PAT stabilisce la corrispondenza fra un programma ed il PID dove viaggiano TSps (Transport Stream packets) che trasportano la definizione di questo programma (il PMT_PID). Ogni TS deve contenere una PAT valida e completa, e mai criptata.

Program Map Table

La PMT stabilisce una mappa di relazioni fra il programma e gli elementi che lo compongono.

L'Mpeg-2 Systems richiede alla PMT una identificazione minima del programma: program_number, PCR_PID, tipi di streams ed elementi di programma. Quando si desidera o si necessita di ulteriori informazioni si deve ricorrere all'utilizzo di descrittori (descriptors), come vedremo più avanti.

I TSps che contengono la PMT devono essere trasmessi in chiaro (senza criptaggio).

Condicional Access Table

La CAT stabilisce l'associazione fra uno o più sistemi d'accesso condizionato, i relativi EMM streams e qualsiasi altro parametro speciale associato ad essi.

Network Information Table

I contenuti della NIT sono privati e non vengono specificati dall'Mpeg-2 Systems. In generale conterranno le mappe dei servizi messe in relazione agli identificatori di TS, frequenze di canale, numeri di transponder del satellite, caratteristiche di modulazione, etc...

Private Sections

Insieme alle tavole PSI è possibile trasportare anche dati privati. Il formato con cui vengono trasportati i dati privati all'interno dei TSps (Transport Stream packets) non è definito dall'Mpeg-2 Systems. Lo

stesso standard offre però la possibilità di strutturare questi dati nello stesso modo in cui lo sono le tavole PSI. Per questo motivo è stata definita una sezione privata (private_section).

Un esempio del suo utilizzo potrebbe essere il trasporto dei dati supplementari che possono essere necessari alle applicazioni che vadano "muntades" (?) nel Mpeg-2.

Descriptors

Se le informazioni trasmesse dalle Tavole PSI non sono sufficienti o si desidera inviare delle informazioni addizionali, si ricorre all'utilizzo di descrittori. I descrittori sono un insieme di strutture sintattiche che permettono il trasporto di informazioni addizionali in modo standardizzato. Esistono diversi descrittori ciascuno diretto a fornire informazione (descrivere) su una qualche struttura o aspetto del Transport Stream, per esempio: un programma, un elementary streams, il sistema di criptaggio, etc... La sua localizzazione è sempre all'interno dei loop delle Sezioni PSI, Benché se si desidera si possono inserire anche all'interno di sezioni private.

| Descrittori definiti dall'Mpeg-2 | |
|-----------------------------------|---|
| Video stream descriptor | System clock descriptor |
| Audio stream descriptor | Multiplex buffer utilization descriptor |
| Hierarchy descriptor | Copyright descriptor |
| Registration descriptor | Maximum bitrate descriptor |
| Data stream alignment descriptor | Private data indicator descriptor |
| Target background grid descriptor | Smoothing buffer |
| Video window descriptor | STD descriptor |
| Conditional access descriptor | IBP_descriptor |
| ISO 639 language descriptor | |

2.3.2. Funzionamento della PSI

Un sistema di comunicazione, in special modo nelle applicazioni "broadcast", può offrire all'ingresso di un ricevitore differenti TSs, ciascuno identificato da un transport_stream_id. Questi transport streams possono arrivare via satellite, cable o rete terrestre. Ogni TS deve presentare un minimo di dati PSI. Sempre deve essere presente una versione completa della PAT, dove vengono elencati tutti i programmi del TS, ed una versione completa della TS_MAP, che contiene definizioni complete di tutti i programmi del TS. Se qualche stream è criptato, allora deve essere presente anche una CAT, che specifichi i relativi EMM Streams (Entitlement Management Messages). La presenza di una NIT è totalmente opzionale.

Ogni TS viene identificato tramite un transport_stream_id e ogni programma con un program_number.

Per poter decodificare un programma, il decoder dovrà conoscere il program_number ed il transport_stream_id del Transport Stream dove questo programma si trova. Dei dati necessari per permettere al decodificatore di sintonizzare questo TS non se ne occupa l'Mpeg-2 ma, come vedremo più avanti, è compito del DVB-SI.

Una volta sintonizzato il TS, il decodificatore cercherà tutti i TSps con PID 0000 e 0001. I pacchetti con PID=0000 contengono i dati PAT e i pacchetti con PID=0001 contengono i dati CAT, necessari se il programma in questione è criptato. Attraverso la lettura della PAT potrà conoscere quali sono i PIDs corrispondenti alla NIT e alle tavole PMT trasportate, una per ogni programma. Selezionerà quindi i pacchetti con il PID che contiene i dati della PMT associata al program_number desiderato (nella Immagine 2.3-1, per il program_number=1 avremo un PID_PMT=22).

Attraverso la lettura della PMT otterrà i PIDs corrispondenti al video (PID=54), audio (PID=48,49..) i dati (PID=66) che compongono il programma ed il PID dei pacchetti che contengono il Program Clock Reference del programma (PID=31).

Con tutta questa informazione il decodificatore sarà già in grado di selezionare i TSps necessari per poter ricostruire gli ESs che compongono il programma.

PSI Table structure

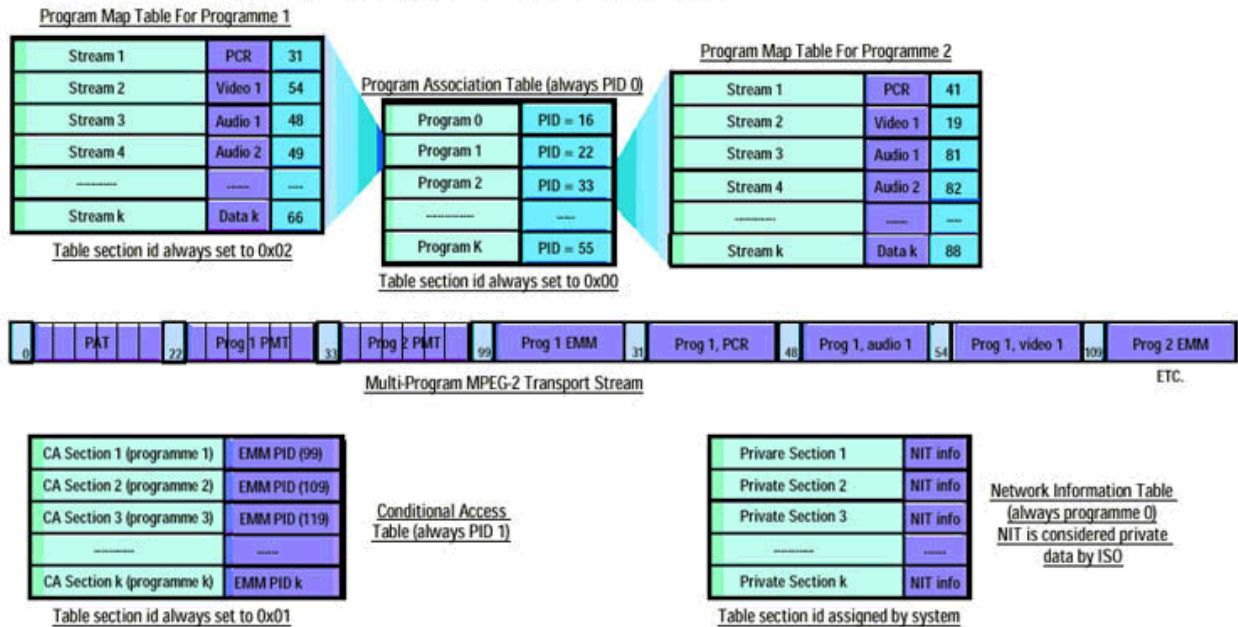


Immagine 2.3-1

Tutto questo processo che abbiamo appena spiegato in termini molto schematici e generali, richiede un certo tempo. Questo tempo viene denominato tempo d'accesso aleatorio (Random Access) ed è caratteristica propria dei sistemi digitali che usano multiplexaggio temporale. Il decodificatore necessiterà di un certo tempo per poter conoscere il contenuto di tutti i TSs e la situazione di tutti gli elementi costituenti un programma. Questo tempo dipenderà dalla frequenza con cui vengono inviate le tavole PSI.

2.3.3. Accesso Aleatorio

Nei sistemi dove l'accesso aleatorio (Random Access) è una considerazione, si raccomanda di ritrasmettere le sezioni PSI ripetutamente. I decoders necessitano dei dati PSI per identificare i contenuti del TS, per poterlo decodificare. L'Mpeg-2 Systems non dice niente riguardo alla frequenza di trasmissione delle tavole PSI. Una frequenza alta, oltre ad accelerare il processo d'accesso aleatorio, provocherà anche un aumento nel bitrate utilizzato per i dati PSI. Se i contenuti di un TS non variano costantemente, i dati PSI possono essere memorizzati dal decoder per permettere un accesso più rapido. La relazione fra la capacità di memoria del decoder ed il bitrate PSI necessario è a discrezione del progettista.

2.3.4. Metodo di trasporto delle tavole PSI

Le tavole elencate precedentemente sono concettuali in quanto mai necessitano di essere regenerate in modo specifico all'interno di un decodificatore. Prima di essere inviate nei TSps possono essere segmentate in sezioni. Successivamente queste sezioni sono "mappate" nel payload dei TSps.

Questo è lo stesso metodo si usa anche per i dati privati, ciò permette di ridurre i costi di progetto. Pertanto ogni tavola PSI non è un insieme continuo di dati ma dati che vengono organizzati in sezioni.

Tutte le sezioni, appartengono a una tavola, condividono una serie di campi che permettono la sua identificazione univoca:

- **Table_id**

L'identificatore di tavola identifica a quale tavola appartiene la sezione.

- Le sezioni con table_id 0x00 appartengono alla PAT
- Le sezioni con table_id 0x01 appartengono alla CAT
- Le sezioni con table_id 0x02 appartengono alla TS PMT

Altri valori del table_id possono essere usati per scopi privati. Il table_id permette una identificazione rapida.

Tavola 2.3-1

| Valore | descrizione |
|-----------|---|
| 0x00 | Program_association_section |
| 0x01 | Conditional_access_section(CA_section) |
| 0x02 | TS_program_map_section |
| 0x03-0x3F | ITU-T Rec. H.222.0 ISO/IEC 13818 reserved |
| 0x40-0xFE | User private |
| 0xFF | Forbidden |

- **Table_id_extension**

Questo campo di 16 bits è presente nelle versioni lunghe di una sezione.

Nella PAT, questo campo si utilizza per identificare il transport_stream_id dello stream (etichetta definita dall'utente, che permette di distinguere un TS dagli altri facenti parte di un network o fra i networks). Questo campo non ha attualmente nessun significato nella CAT. In una TS Program Map Section il campo contiene il program_number, che identifica il programma al quale i dati della sezione si riferiscono.

- **Section_number**

Questo campo permette alle sezioni di una certa tavola di essere raggruppate dal decodificatore nell'ordine originale. Mpeg-2 Systems non stabilisce che le sezioni debbano essere trasmesse seguendo l'ordine numerico, ciononostante è raccomandato a meno che si desideri trasmettere alcune sezioni della tavola più frequentemente che altre, per esempio per considerazioni riguardanti l'accesso aleatorio.

- **Version_number**

Quando le caratteristiche del TS descritte dalla PSI cambiano (per esempio: vengono aggiunti programmi, si modificano gli ES che formano un certo programma...), allora si devono inviare nuovi dati PSI con le informazioni aggiornate e indicando che sono la versione trasmessa più recente. I decoders devono essere capaci di identificare se la sezione ricevuta più recente è identica alla sezione che han già processato/immagazzinato

(in questo caso la sezione è scartata) o se è differente, e può, quindi, significare un cambio di configurazione. Il cambio di configurazione si ottiene attraverso l'invio della sezione in questione, con i dati aggiornati e con il campo `version_number` incremento di una unità rispetto alla sezione precedente.

- **Current_next_indicator**

È importante conoscere in ogni istante quali dati PSI sono validi e quali no. Perciò ogni sezione può essere marcata come quella attualmente valida (`current`), o come quella valida in un futuro immediato (`next`). Ciò permette la trasmissione di configurazioni future e così dare al decoder l'opportunità di prepararsi in previsione. Non è obbligatorio trasmettere la versione successiva di una sezione in anticipo però se viene trasmessa deve essere quella corretta.

2.3.5 La "mappatura" delle Sezioni nei Transport Stream Packets

Le tavole PSI sono mappate nei di TSps attraverso la struttura delle sezioni. Le sezioni hanno una lunghezza massima di 1024 bytes per le tavole, e di 4096 bytes per le sezioni private. Sono "mappate" direttamente nei TSps, ciò significa senza una mappatura previa in PESps. Le sezioni non iniziano necessariamente all'inizio dei TSps, giacché l'inizio della prima sezione nel payload di un TSp è indicato attraverso un puntatore: `pointer_field`. Il `payload_unit_start_indicator` vale '1' nei pacchetti PSI in cui è presente il `pointer_field`. Il `pointer_field` informa su dove è collocato l'inizio della prima sezione all'interno del pacchetto (TSp). Non c'è mai più di un `pointer_field` in un TSp. L'inizio di qualsiasi altra sezione può essere localizzato contando la lunghezza della prima e sommando la lunghezza delle seguenti sezioni, dato che non ci possono essere spazi vuoti fra sezioni all'interno di un TSp.

È importante notare che all'interno dei TSps di qualsiasi PID concreto, una sezione deve terminare prima che la seguente possa essere inviata, altrimenti non sarebbe possibile identificare a quale sezione appartengono i dati. Se una sezione termina prima della fine di un TSp e non conviene iniziare ad inviarne una nuova, si ricorre ad un meccanismo di riempimento per finire di riempire il pacchetto.

Il valore dei bytes di riempimento è '0xFF', i vengono collocati dopo l'ultimo byte di una sezione ed arrivano fino alla fine TSp.

Ogni sezione ha un campo `table_id` nel proprio header. Questo campo permette alle sezioni delle tavole PSI e dati privati di essere inviate in pacchetti (TSps) con lo stesso stesso PID. Quando si tratta di sezioni TS PMT (o NIT) i dati privati possono mescolarsi dentro uno stesso TSp. Le sezioni private non possono però venir mappate all'interno di pacchetti PAT o CAT.

Tutte le sezioni PAT sono mappate in TSps con PID=0x0000 e tutte le sezioni CAT in pacchetti con PID=0x0001.

Le sezioni PMT possono essere mappate dentro pacchetti con un PID scelto dall'utente, e indicato come `PMT_PID` per ogni programma nella PAT. Ugualmente, il PID dei pacchetti che contengono la NIT può essere scelto dall'utente, devono però essere indicati dalla PAT con il `program_number = 0x00`, se la NIT esiste.

Le tavole private possono essere trasmesse utilizzando la sintassi della `private_section()`. Queste tavole possono essere usate per esempio in ambienti di *broadcasting* per descrivere un servizio, un evento che deve essere inviato, la guida dei programmi e informazioni correlate (vedere punto 3).

2.3.6. Program Association Table

La Program Association Table fornisce la corrispondenza esistente fra un program_number (etichetta numerica associata ad un programma) ed il valore del PID dei pacchetti che trasportano la definizione di questo programma (il PMT_PID).

Ogni TS deve contenere una PAT valida, completa e mai criptata.

I pacchetti che trasportano dati che fanno parte della PAT, devono avere un PID di valore '0x000' devono insieme devono fornire una lista completa di tutti i programmi che formano il TS.

Qualsiasi cambio nei programmi trasportati dal TS devono essere descritti in una versione aggiornata della PAT trasportata nei TSps di PID '0x0000'.

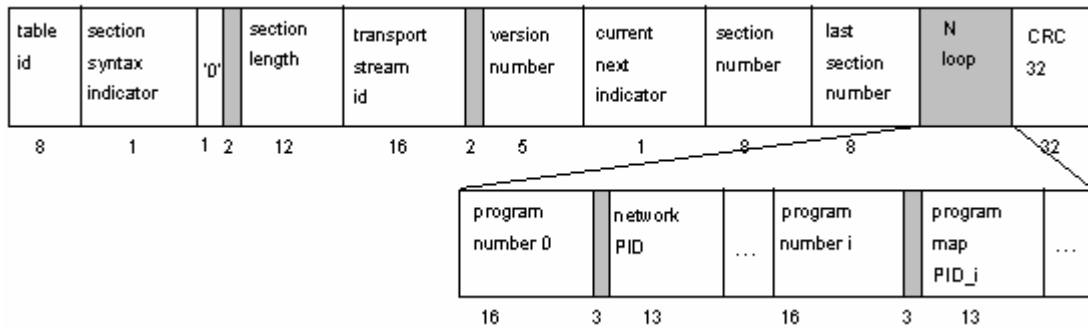
La versione trasmessa più recente della tavola con current_next_indicator a '1' deve essere sempre applicabile sempre ai dati attuali del TS.

La PAT può essere divisa in 255 sezioni al massimo prima di essere "mappata" nei TSps. Ogni sezione trasporta una parte della totalità della PAT.

Questa divisione può essere opportuna per minimizzare la perdita di dati in condizioni d'errore. Se un pacchetto va perso o c'è bits erroneo in una sezione piccola della PAT, il resto delle sezioni possono essere correttamente decodificate. Se tutte le informazioni della PAT stessero dentro una sola sezione, un errore che causi un cambio in un bit del table_id, per esempio, causerebbe la perdita della totalità della PAT. In ogni caso, ciò è permesso se la sezione osserva il limite di 1024 bytes.

Tavola 2.3-2 Program association section

| Sintassi | Numero di bits |
|---------------------------------|----------------|
| Program_association_section() { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| '0' | 1 |
| reserved | 2 |
| section_length | 12 |
| transport_stream_id | 16 |
| reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| for (i=0; i<N;i++) { | |
| program_number | 16 |
| reserved | 3 |
| if(program_number == '0') { | |
| network_PID | 13 |
| } | |
| else { | |
| program_map_PID | 13 |
| } | |
| } | |
| CRC_32 | 32 |
| } | |



2.3.7 Definizione semantica dei campi della Program Association Section

table_id è un campo di 8 bits, il valore del quale deve essere '0x00'.

Section_syntax_indicator È un campo di un bit che deve valere '1'.

Section_length È un campo di 12 bits. I primi due bits devono valere '0'. I 10 bits rimanenti specificano il numero di bytes della sezione a partire da questo campo, includendo il CRC. Questo campo non può eccedere il valore 1021(0x3FD).

Transport_stream_id È un campo di 16 bits che funziona da etichetta per identificare questo TS da qualsiasi altro multiplexato all'interno di un network. Il suo valore è definito dall'utente.

Version_number È un campo di 5 bits, il valore del quale indica la versione di tutta la PAT. Il version_number deve essere incrementato di una unità sempre che le informazioni della PAT cambino. Quando il current_next_indicator vale '1', il version_number deve essere quello che appartiene alla versione della PAT attualmente applicabile. Quando il current_next_indicator vale '0' il version_number deve essere quello che corrisponde alla prossima versione della PAT applicabile.

Current_next_indicator È un indicatore di un bit. Se vale '1' indica che la PAT inviata è attualmente applicabile. Quando vale '0', indica che la tavola inviata non è ancora applicabile e deve essere la prossima tavola diventare valida.

Section_number È un campo di 8 bits, il valore del quale è il numero di questa sezione. Il section_number della prima sezione della PAT deve essere '0x00' e deve essere incrementato di una unità per ogni sezione addizionale della PAT.

Last_section_number È un campo di 8 bits il valore del quale indica il numero dell'ultima sezione di tutta la PAT.

Program_number È un campo di 16 bits. Specifica il programma al quale è applicabile il program_map_PID. Permette la definizione di 65535 programmi in un TS. Il valore '0x0000' è riservato alla tavola NIT ed il seguente PID deve essere il network_PID, in tutti gli altri casi il valore di questo campo è definito dall'utente. Non può acquisire lo stesso valore più di una volta all'interno di una stessa versione della PAT.

Nota: Il program_number per esempio può essere usato per designare un canale di diffusione (broadcast).

Network_PID È un campo di 13 bits che è usato soltanto quando il valore del program_number vale '0x0000'. Especifica il PID dei pacchetti che contengono la NIT. Il valore del network_PID è definito dall'utente, però può prendere soltanto i valori specificati nella tavola 2.2.4-9.

Program_map_PID (PMT_PID) È un campo di 13 bits che stabilisce il PID dei TPps che contengono la program_map_section applicabile al programma specificato dal program_number. Nessun program_number può avere più di un program_map_PID assegnato. Il valore del program_map_PID è definito dall'utente, però può prendere soltanto i valori specificati nella tavola 2.2.4-9.

CRC_32 È un campo di 32 bits che contiene il valore di CRC da applicare alla totalità della program_association_section.

2.3.8. Conditional Access Table

La CAT stabilisce le associazioni fra uno o più sistemi d'accesso condizionato, i suoi EMM streams e qualsiasi altro parametro speciale associato.

In qualsiasi TS nel quale uno o più ES sia criptato, i TSps trasmessi con PID '0x0001' devono contenere una Conditional Access table completa, inclusi i CA_descriptors associati agli streams criptati.

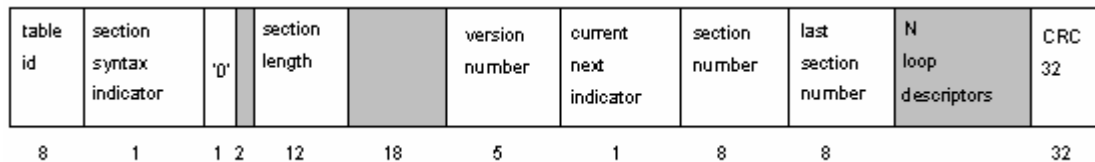


Immagine 2.3-3

Qualsiasi cambio nel sistema di criptaggio che renda non più valida la tavola esistente deve essere riportato in una versione aggiornata della CAT.

Prima di essere inserita all'interno dei TSps, la tavola viene segmentata in una o più sezioni, applicando la seguente sintassi.

Tavola 2.3-3 Conditional access section

| Sintassi | Numero di bits |
|---------------------------------|----------------|
| CA_section() { | |
| Table_id | 8 |
| Section_syntax_indicator | 1 |
| '0' | 1 |
| Reserved | 2 |
| Section_length | 12 |
| Reserved | 18 |
| Version_number | 5 |
| Current_next_indicator | 1 |
| Section_number | 8 |
| Last_section_number | 8 |
| For (i=0; i<N;i++) { | |
| descriptor() | |
| } | |
| CRC_32 | 32 |
| } | |

2.3.9. Program Map Table

La PMT fornisce una mappa di relazioni fra il program_number i gli elementi lo compongono. I TS packets che trasportano questa tavola non possono mai essere criptati e corrispondono ad uno o più PIDs selezionati "privatamente". È possibile trasportare dentro uno stesso TSp sezioni TS PMT che si riferiscano a programmi differenti.

Tutti i programmi sono elencati nella PAT e ciascuno è descritto all'interno di una unica TS_program_map_section. Pertanto il campo section_number vale sempre '0', poichè per un programma non c'è mai più di una sezione.

Qualsiasi cambio nella definizione di qualsiasi programma trasportato dal TS deve essere descritto in una versione aggiornata della corrispondente sezione della PMT e la PMT deve essere trasportata nei pacchetti di PID identificato come il program_map_PID per questo programma specifico.

Tutti i pacchetti che trasportano una TS_program_map_section determinata, hanno lo stesso valore di PID, e durante l'esistenza continua di un programma, includono tutti gli eventi associati, il program_map_PID non può cambiare.

Ogni sezione dunque si occupa della definizione di un solo programma.

Tavola 2.3-4 Transport Stream program map section

| Sintassi | Numero di bits |
|---------------------------------|----------------|
| TS_program_map_section() { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| '0' | 1 |
| Reserved | 2 |
| section_length | 12 |
| program_number | 16 |
| Reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| Reserved | 3 |
| PCR_PID | 13 |
| Reserved | 4 |
| program_info_length | 12 |
| for (i=0; i<N; i++) { | |
| descriptor() | |
| } | |
| for (i=0; i<N1; i++) { | |
| stream_type | 8 |
| Reserved | 3 |
| elementary_PID | 13 |
| Reserved | 4 |
| ES_info_length | 12 |
| for (i=0; i<N2; i++) { | |
| descriptor() | |
| } | |
| } | |
| CRC_32 | 32 |
| } | |

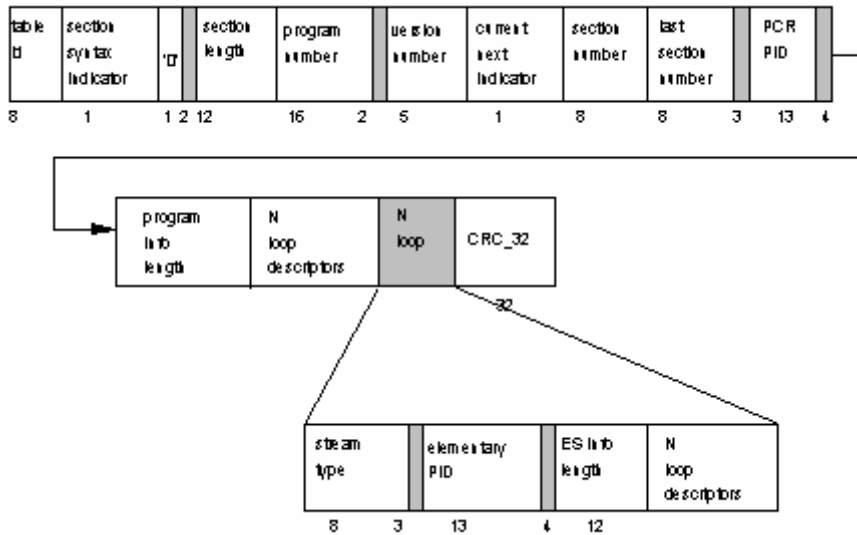


Immagine 2.3-4

I descrittori inclusi nel primo loop (dopo del campo program_info_length) forniscono informazioni sul programma, e quelli inclusi nel terzo loop (dopo del campo ES_info_length) fanno riferimento agli ES.

2.3.10 Definizione semantica dei campi del Transport Stream Program Map Section

Program_number È un campo di 16 bits. Specifica il programma al quale è applicabile il program_map_PID. Una descrizione di programma deve essere trasportata dentro una sola TS_program_map_section. Ciò implica che una definizione di programma non possa mai essere di lunghezza superiore a 1016(0x3F8) bytes.

Section_number Il valore di questo campo di 8 bits deve essere '0x00'.

Last_section_number Il valore di questo campo di 8 bits deve essere '0x00'.

PCR_PID È un campo di 13 bits che indica il PID dei TSps che contengono i campi PCR validi per il programma specificato dal program_number.

Stream_type È un campo di 8 bits che stabilisce il tipo di elemento trasportato dai pacchetti aventi il PID definito nel campo elementary_PID. I valori dello stream_type sono specificati nella tavola 2.3-5

Tavola 2.3-5 Stream type assignments

| Valore | Descrizione |
|-----------|--|
| 0x00 | ITU-T ISO/IEC Reserved |
| 0x01 | ISO/IEC 11172 Video |
| 0x02 | ITU-T Rec. H.262 ISO/IEC 13818-2 Video or ISO/IEC 11172-2 constrained parameter video stream |
| 0x03 | ISO/IEC 11172 Audio |
| 0x04 | ISO/IEC 13818-3 Audio |
| 0x05 | ITU-T Rec. H.222.0 ISO/IEC 13818-1 private_sections |
| 0x06 | ITU-T Rec. H.222.0 ISO/IEC 13818-1 PES packets containing private data |
| 0x07 | ISO/IEC 13522 MHEG |
| 0x08 | ITU-T Rec. H.222.0 ISO/IEC 13818-1 Annex A DSM CC |
| 0x09 | ITU-T Rec. H.222.1 |
| 0x0A | ISO/IEC 13818-6 type A |
| 0x0B | ISO/IEC 13818-6 type B |
| 0x0C | ISO/IEC 13818-6 type C |
| 0x0D | ISO/IEC 13818-6 type D |
| 0x0E | ISO/IEC 13818-1 auxiliary |
| 0x0F-0x7F | ITU-T Rec. H.222.0 ISO/IEC 13818-1 Reserved |
| 0x80-0xFF | User Private |

Elementary_PID È un campo di 13 bits che stabilisce il PID dei pacchetti (TSps) nei quali viene trasportato l'elemento associato.

ES_info_length È un campo di 12 bits, i primi 2 bits devono essere '00'. I 10 bits rimanenti specificano il numero di bytes dei descrittori, associati all'elemento di programma, che cominciano immediatamente dopo del campo ES_info_length.

2.3.11. Sintassi della Private Section

Insieme alle tavole PSI è possibile trasportare tavole di dati privati. Il formato attraverso il quale vengono trasportati dati privati dentro ai TSps non è definito dall' MPEG-2 Systems. Lo standard offre però la possibilità che questi dati siano strutturati nello stesso modo che quelli che trasportano le tavole PSI. La mappatura dei dati privati è identica a quella delle tavole PSI. Per questo motivo è stata definita la sezione privata (private_section).

La private_section può essere usata in due modi: se il section_syntax_indicator vale '1', sono presenti tutti i campi comuni a tutte le tavole, si chiama versione corta; se l'indicatore vale '0', sono presenti soltanto i campi fra il table_id ed il private_section_length (incluso), ed il resto dei bytes della private_section possono avere qualsiasi struttura determinata dall'utente.

Il numero massimo di bytes in una private_section è di 4096 bytes, e soltanto i dati privati, che occupano i bytes private_data_byte, possono essere criptati, il resto dei campi non possono essere criptati.

Le Private_sections possono essere trasportate in pacchetti (TSps) con valori PIDs esclusivamente assegnati a sezioni private (includendo il NIT_PID) o anche con valori assegnati nella PMT (PID=PMT_PID). Se il PID di questi TSps è definito nella PAT esclusivamente come PID di sezione privata (corrisponde a stream_id=0x05), allora le sezioni private possono essere trasportate solo dentro TS packets con questo valore di PID.

Si può formare una tavola privata attraverso varie private_sections con lo stesso valore di table_id.

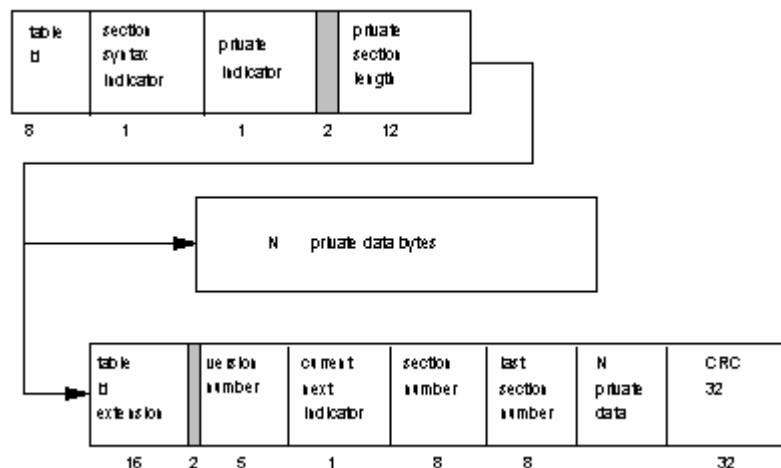


Immagine 2.3-5

Tavola 2.3-6 Private section

| Sintassi | Numero di bits |
|---|----------------|
| private_section() { | |
| Table_id | 8 |
| Section_syntax_indicator | 1 |
| Private_indicator | 1 |
| Reserved | 2 |
| Private_section_length | 12 |
| If (section_syntax_indicator == '0') { | |
| for (i=0;i<N;i++) { | |
| private_data_byte | 8 |
| } | |
| } | |
| Else { | |
| table_id_extension | 16 |
| Reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| for (i=0;i<private_section_length-9;i++) { | |
| private_data_byte | 8 |
| } | |
| CRC_32 | 32 |
| } | |
| } | |

2.3.12 Definizione semantica dei campi della private section

Section_syntax_indicator È un indicatore di un bit. Quando vale '1' indica che la private_section segue la sintassi di sezione generica a partire del campo private_section_length. Quando vale '0' indica che i private_data_bytes sono collocati immediatamente dopo il campo private_section_length.

Private_indicator È un campo di un bit definibile dall'utente ed il suo uso non sarà specificato dalla ITU-T Rec. H.222.0 | ISO/IEC in futuro.

Private_data_byte Il campo private_data_byte è definibile dall'utente ed il suo uso non sarà specificato dalla ITU-T Rec. H.222.0 | ISO/IEC in futuro.

Table_id_extension È un campo di 16 bits. Il suo uso e valore è definibile dall'utente.

2.3.13. Network Information Table

La Network Information Table è opzionale ed i suoi contenuti sono privati. In linea generale conterranno dettagli non soltanto del TS che il trasporta, ma anche di altri TSs che possono essere disponibili nello stesso decoder, come per esempio, `transport_stream_ids`, frequenze di canale, numeri di transponder del satellite, caratteristiche di modulazione, etc...

Se è presente, deve essere obbligatoriamente strutturata in una o più `Private_Sections`, ed è trasportata in pacchetti con un unico valore di PID, il `network_PID`. Questo valore è definito dall'utente nella PAT, e deve essere associato al `program_number` di valore '0x0000', che è riservato per questo uso.

Nel punto 3.3 si illustra quali sono i contenuti che il DVB-SI specifica, e come devono essere strutturati per questo caso in particolare.

2.4. Decodificatore (Decoder)

Come già abbiamo avuto modo di dire, l'Mpeg-2 è uno standard generico, pensato per un grande numero di applicazioni con requisiti molto diversi. È per questo motivo che non stabilisce in nessun momento un modello di codificatore (encoder) né di decodificatore (decoder), e lascia nelle mani dei progettisti e dei produttori la sua progettazione. Con ciò, si suppone l'esistenza di differenti modelli più specializzati ed ottimali per le diverse applicazioni, e la possibilità di ampliare il raggio di applicabilità dell'Mpeg-2 parallelamente allo sviluppo di nuove applicazioni, ed il continuo miglioramento dei progetti esistenti.

Quello che invece viene fornito dall'Mpeg-2 è un modello matematico di decoder ideale che si chiama System Target Decoder (STD), e concretamente per quanto concerne il TS prende il nome di Transport Stream System Target Decoder (T-STD).

Il T-STD nasce dalla necessità dell'Mpeg-2 Systems di definire in modo preciso il movimento di tutti i bits dentro i fuori dai buffers, i processi di decodifica e gli istanti in cui queste operazioni avvengono. Per cui, l'SDT-T è un modello concettuale del decodificatore ideale, nel quale i buffers non subiscono mai uno svuotamento o un sovraccarico, e pertanto serve da riferimento e punto di partenza per i produttori.

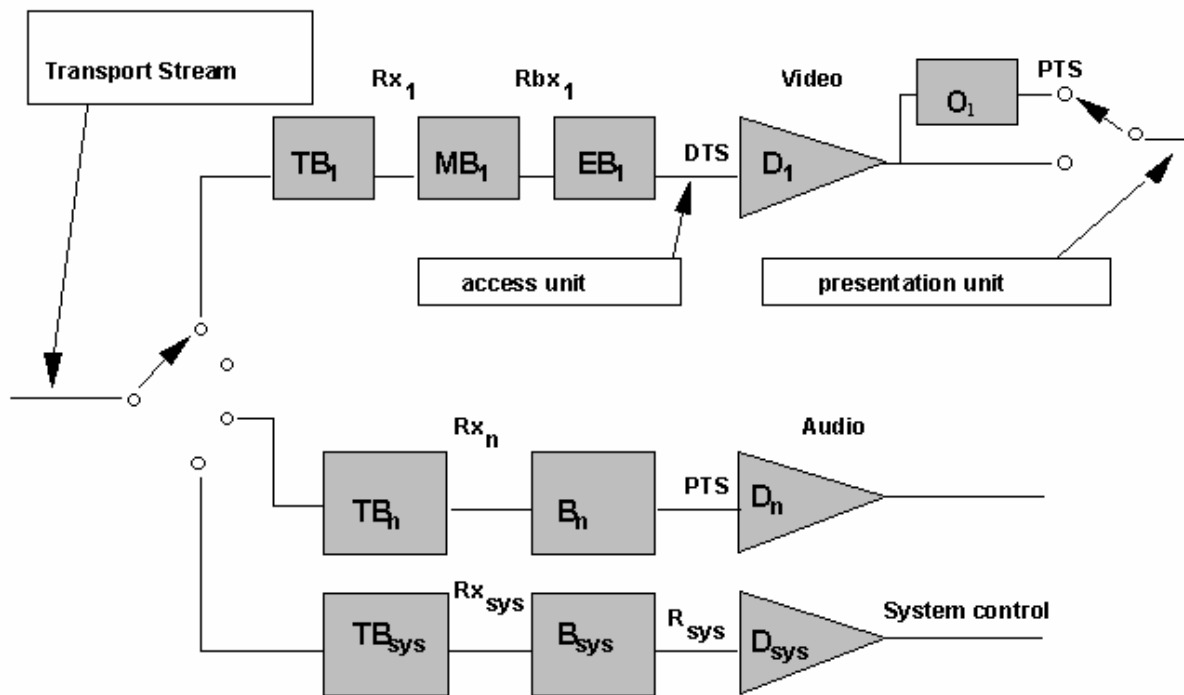
L'STD è una semplificazione rispetto alle implementazioni reali e fisiche di decoders; questo per rendere più chiaro il suo funzionamento. Due semplificazioni importanti sono:

- A) Suppone che l'STC sia sempre perfettamente sincronizzato con l'STC del programma che si vuole decodificare, ciò significa che non contempla la rigenerazione del clock.
- B) Non tiene conto dei ritardi e dei tempi di risposta dei dispositivi reali, considera tutti i processi istantanei.

Noi sfrutteremo il T-STD per spiegare il suo funzionamento e capire meglio la struttura Transport Stream. Cercheremo di non entrare in dettagli troppo concreti e tecnici e di fare alcune considerazioni generali per le implementazioni reali.

Occorre ricordare sempre che il T-STD è semplicemente una raccomandazione che i fabbricanti possono adottare o no.

2.4.1. Transport Stream System Target Decoder



Imatge 2.4-1

Questo è lo schema del T-STD.

L'elemento di partenza del T-STD è il Transport Stream (TS). Un TS può essere formato da molteplici programmi con basi di tempo indipendenti (PCR's indipendenti). In ogni caso il T-STD decodifica soltanto un solo programma per volta e tutte le indicazioni di tempo che utilizza fanno riferimento alla base di tempo del programma che si sta decodificando.

Come già abbiamo avuto modo di vedere un programma è un insieme di ESs che hanno una base di tempo comune, e che sono pensati per una decodifica e una presentazione sincronizzata. La sua definizione avviene attraverso la Program Map Table della PSI, ed è a partire delle informazioni che questa tavola fornisce che il demultiplexer che sta all'ingresso del T-SDT seleziona i TS packets che contengono gli n ESs che costituiscono il programma desiderato.

La totalità dei TS packets selezionati vengono immagazzinati nei *buffers di trasporto* (TB). A partire dal PID possiamo sapere di quale tipo di ES si tratta e dirigere i packets verso i buffers audio o video. I TS packets che contengono dati della PAT, CAT e PMT del programma in questione, vengono immagazzinati nell' *buffers di trasporto del sistema*.

I *buffers di trasporto* hanno una dimensione di 512 bytes ed i dati vi entrano a seconda del *transport_rate* del TS, che si può determinare a partire dall'equazione 2.2.3-4.

In seguito, tutti i bytes che fanno parte di un PES packet o del suo contenuto, vengono passati ad un secondo insieme di buffers. Questo trasferimento viene eseguito fa ad un bitrate R_x che dipende dal tipo di dati, e nel caso del video dipende anche dal livello e dal profile (vedere tavola 2.4-1).

I PES packets dell'audio e quelli di sistema, vengono passati ai *buffers principali* (B). I PES packets del video vengono invece passati ai *buffers di multiplexaggio* (MB). La capacità di tutti questi buffers si può vedere nella tavola 2.4-2.

Ci concentriamo ora sul cammino dei dati relativi al video. I *buffers di multiplexaggio* hanno l'obiettivo di alleviare gli effetti del multiplexaggio dei TS packets. Si specificano due metodi per trasferire il payload di ogni PES packet, cioè gli ES del video, ai successivi buffers: i *buffers principali* (EB). Questi due metodi sono il *Leak method* ed il *Vbv delay method*.

Tavola 2.4-1

| Transport Stream System Target Decoder (TS-STD) transfer rates for Main Profile Video,* Audio, and Systems streams. | | |
|---|----------------|------------|
| Elementary Stream Rate (Rbx) | TB Rate (Rx) | MB |
| Low Level Video | 4.8 Mbits/s | 4 Mbits/s |
| Main Level Video | 18 Mbits/s | 15 Mbits/s |
| High 1440 Level Video [I.05xbit-rate, 60Mbs] | 72 Mbits/s MIN | |
| High Level Video [I.05xbit-rate, 80Mbs] | 96 Mbits/s MIN | |
| Audio n.a. | 2 Mbits/s | |
| Systems data n.a. | 1 Mbits/s | |

*The parameters bit rate and vbv-buffer_size are given in the video stream.

Tavola 2.4-2

| Transport Stream-System Target Decoder (TS-STD) buffer sizes for Main Profile Video, Audio and Systems streams. | | | |
|---|-----------|------------------|-------------------------|
| Elementary Stream | TB Size | MB Size | EB Size |
| Low Level Video | 4096 bits | 496 469 bits | EB size vbv_buffer_size |
| Main Level Video | 4096 bits | 1 915 008 bits - | EB size vbv_buffer_size |
| High 1440 Level Video | 4096 bits | 320 000 bits | vbv_buffer_size |
| High Level Video | 4096 bits | 426 667 bits | vbv_buffer_size |
| Audio | 4096 bits | n.a. 28 | 672 bits |
| Systems data | 4096 bits | n.a. 12 | 288 bits |

Il *Leak method* si può intendere come il metodo normale e viene applicato quando il trick mode è 'vero', quando il descriptor SDT non è presente, e se lo è quando esso lo indichi. Il metodo semplicemente stabilisce che quando il buffer EB non è pieno, i dati dell'MB saranno trasferiti con un bitrate *Rbx* (come si può vedere nella tavola 2.4-1).

Il *Vbv delay method* è più complesso e viene applicato al resto dei casi. Fondamentalmente la sua necessità sorge nelle sequenze *low_delay*. Queste sono più suscettibili (a causa del fatto che non fanno uso di immagini B) ad aver bisogno di una "gran quantità" di bytes per codificare alcune immagini, quelle chiamate *Big Pictures*. La totalità di una di queste immagini ci mette più tempo ad essere ricevuta, e ciò può comportare un ritardo non desiderato rispetto all'istante di decodifica indicato dal DTS. Il metodo si basa nell'indicare attraverso il campo *vbv_delay* (contenuto nella *picture_header*) l'istante fino al quale si devono attendere i bytes della *big picture* nel buffer MB, per dopo essere passati con un bitrate *Rbx*

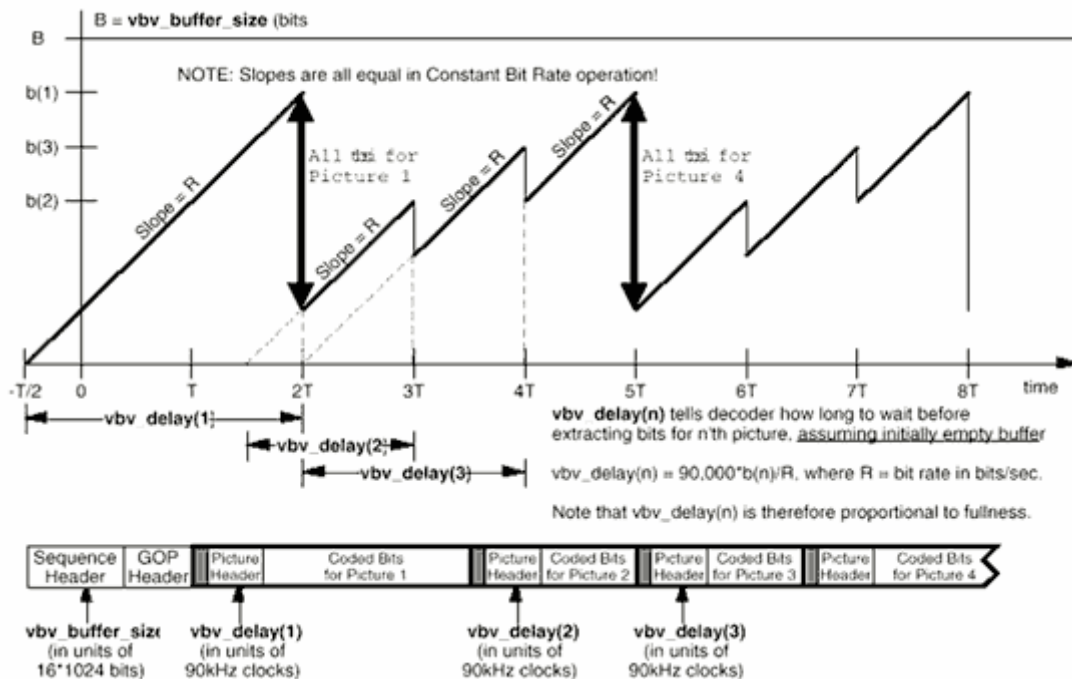


Immagine 2.4-2

al buffer EB. Questo istante è: DTS meno vbv_delay . Fin qui sembrerebbe che non ci possano essere contrattimpim, in quanto i bytes vengono passati al buffer EB prima dell'istante di decodifica DTS. Però il problema è che questo processo avviene ad un bitrate uguale o inferiore al bitrate R_{bx} del metodo *Leak method*, e pertanto può accadere che quando arrivi l'istante DTS, la totalità della *big picture* non sia trasferita al buffer EB e si debba continuare a mostrare l'immagine precedente. Ciò accadrebbe fino a che la totalità della *big picture* non fosse stata interamente trasferita e potesse quindi essere decodificata. D'altra parte, il codificatore dovrà incaricarsi di omettere la codifica delle immagini necessarie per riparare al ritardo provocato nella sequenza.

La capacità del buffer EB viene determinata dal campo vbv_buffer_size (contenuto nella *sequence_header*) come si può vedere nella tavola 2.4-2.

$$R_{bx}(j) = NB(j) / (vbv_delay(j) - vbv_delay(j+1) + tdn(j+1) - tdn(j))$$

dove $NB(j)$ è il numero di bytes fra gli ultimi bytes dei codici di inizio immagine (incluso l'ultimo byte del secondo codice d'inizio) delle immagini j e $j+1$, escludendo i bytes dello header dei PESps.

Tutti i bitrates e le capacità dei buffers vengono calcolati in modo che in nessun momento (ad eccezione di casi speciali, p.e. Vbv_delay) nessuno dei buffers possa subire un sovraccarico o uno svuotamento e la conseguente perdita d'informazione.

Ci troviamo ora in un punto dove tutti i dati stanno nei buffers principali.

I dati di sistema sono decodificati ad un bitrate R_{sys} che aumenta all'aumentare del $transport_rate$, ciò permette rates più elevati per i dati PSI a seconda del $transport_rate$.

$$R_{sys} = \max[80\,000\,bits/s, transport_rate(i) \times 8\,bits/byte \div 500]$$

Nelle implementazioni reali i dati decodificati vengono conservati in un memoria e utilizzati per il controllo del sistema.

Le unità d'accesso audio, ancora incapsulate in pacchetti PES packets, attendono l'istante PTS per essere decodificate e presentate, ed una volta giunto questo momento vengono cancellate dal buffer B.

Nel caso del video, gli ES immagazzinati nei buffers EB vengono decodificati nell'istante DTS (o PTS) dai decoders D e cancellati. Successivamente possono essere ritardati nei buffers di riordino (O) prima di essere presentati all'uscita del T-STD. I buffers di riordino sono utilizzati, nel caso degli ES video nei quali le unità d'accesso non sono trasmesse nell'ordine di presentazione. In particolare, se una immagine I (o P) è trasmessa prima di una o più immagini B, allora dovrà essere ritardata nel buffer di riordino, prima di essere presentata e fino a che la successiva immagine I (o P) sia decodificata. Nel frattempo, le successive immagini B sono decodificate e presentate.

Per le unità di presentazione che non richiedono riordino, il PTS coincide con il DTS, visto che nel T-SDT si considera che le unità d'accesso sono decodificate istantaneamente; questo è il caso, per esempio, dei frames B. Per le unità di presentazione che sono ritardate, gli istanti PTS e DTS differiscono di un tempo uguale al ritardo che subisce l'immagine I (o P) nel buffer di riordino, ed è sempre un multiplo del periodo di immagine normale.

I campi PTS e DTS non indicano, di per se stessi, l'occupazione corretta dei buffers del decodificatore in nessun istante.

Esistono però due casi speciali al momento di gestire i buffers EB.

1) In sequenze *low_delay*, se quando arriva l'istante di decodifica DTS di una immagine, non è presente la totalità della unità d'accesso nel buffer EB, di quello che si chiama **Low Delay**.

A partire di questo istante il buffer viene esaminato negli intervalli del periodo di due campi, fino a che l'unità d'accesso può essere decodificata e cancellata dal buffer.

2) Il secondo caso, è nelle immagini del tipi B che utilizzano il Tick mode di pausa, marcia lenta avanti, o marcia lenta indietro. In questi casi l'unità d'accesso B viene mantenuta nel buffer EB fino a che si sia decodificato e presentato per l'ultima volta (vedere punto 2.2.7.1 trick mode).

Nel modello STD le operazioni realizzate vengono considerate istantanee, ad eccezione dei tempi di immagazzinamento nei buffers. In un decodificatore reale esisteranno un insieme di ritardi che dovranno essere tenuti in conto al momento di realizzare il progetto. Per esempio, se le immagini video ci mettono ad essere decodificate esattamente un intervallo di presentazione $1/P$, dove P è il "frame rate", ed i dati video compressi arrivano al decodificatore ad un bit rate R, la conclusione del cancellamento dei bits associati ad ogni immagine subisce un ritardo di $1/P$ rispetto all'istante indicato dai campi PTS e DTS. Ciò comporterà che il buffer del decodificatore video debba essere più grande di una quantità R/P rispetto a quanto specificato nel modello T-STD; cmporterà inoltre che la presentazione del video risulti ritardata rispetto a quella del STD, pertanto i PTS's dovranno essere modificati conformemente. Si il video viene ritardato, la decodifica e la presentazione dell'audio dovrà essere ritardata di una quantità simile, per ottenere una sincronizzazione corretta. Il ritardo della decodifica o presentazione dell'audio del video può essere implementada, per esempio, sommando una costante al valore dei PTSs al loro utilizzo all'interno del decodificatore.

3. DVB-SI

3.1 Introduzione al progetto DVB

Il progetto DVB nasce nel 1993 come risposta alle necessità di fornire un formato comune che permettesse la diffusione della Televisione Digitale direttamente nelle nostre case.

Negli anni 90 era già evidente che la Televisione Digitale avrebbe soddisfatto i requisiti delle aziende fornitrici di servizi, ma che sarebbe stato fattibile soltanto se si fosse adottato uno standard comune a tutti providers europei. L'eredità di molteplici standards del mondo analogico (PAL I, PAL B/G, SECAM, NTSC) doveva essere superata.

Studi economici consigliavano che i nuovi standards fossero adottati rapidamente dall'industria interessata. Il complesso processo che supponevano i sistemi di Televisione Digitale, richiedeva una scala di integrazione dei processori supportabile economicamente soltanto con lo sviluppo di standard comuni. Ciò significa che l'implementazione della Televisione Digitale avrebbe potuto diventare una realtà soltanto se poteva fornire rientri economici che compensassero tutti gli investimenti necessari.

Per assicurare questo, i membri dell'industria europea di radiodiffusione si sono riuniti per pianificare un accordo comune. Hanno dato vita al progetto DVB, basato su un gruppo di lavoro che nel 1990 aveva già fatto uno studio per dimostrare la fruibilità di sistemi di compressione de video digitale.

Dal suo debutto, nel 1993, il progetto è cresciuto fino ad includere rappresentanti di più di 200 aziende di 25 paesi di tutto il mondo. Le sezioni commerciali hanno definito le esigenze del consumatore dal punto di vista dei fornitori di servizi e dei produttori, mentre le sezioni tecniche hanno fornito le soluzioni tecnologiche.

Una delle prime decisioni prese dal progetto DVB è stata quella di adottare l'MPEG-2 come standard di compressione video e audio. Ciò risolveva uno dei requisiti economici di base, l'asimmetria fra l'encoder ed il decoder per abbassare il costo dei decoders che dovranno essere presenti in tutte le famiglie.

Un'altra delle funzioni principali del progetto DVB era di definire tecniche di modulazione e metodi di codifica per la correzione degli errori, che permettessero la trasmissione via satellite, cavo e sistemi di radiodiffusione terrestre (standards DVB-S, DVB-C e DVB-T rispettivamente). Ha fornito anche un algoritmo comune per i sistemi d'accesso condizionato, ha definito la trasmissione delle informazioni di servizio (DVB-SI) che permettono allo spettatore un accesso facile e rapido al programma desiderato ed ha stabilito formati per l'inserimento di dati per applicazioni come i sottotitoli o il teletext.

Il progetto di DVB non stabilisce i propri standards, ma fornisce le specifiche che vengono inviate alle organizzazioni riconosciute che producono standards: ETSI, CENELEC, ITU-R, ITU-T e DAVIC. Fin dal loro inizio gli standards per la trasmissione di DVB via satellite, cavo o sistemi terrestri sono stati elaborati dall'ETSI, e adottati in molte regioni del mondo.

Da parte nostra, concentreremo la nostra attenzione esclusivamente sullo standard DVB Service Information.

3.2. Introduzione alla DVB-Service Information (DVB-SI)

La TV Digitale distribuisce un grande numero di servizi (canali TV, teletexts, emittenti radio, servizi interattivi,..) difficile da memorizzare dal telespettatore. Per cui, sia il telespettatore sia l'Integrated Receiver Decoder (IRD) hanno bisogno di aiuto al momento di selezionare il servizio desiderato e di visualizzare quelli disponibili. Per poter fornire questi meccanismi di aiuto l'IRD ha bisogno di più informazioni rispetto a quelle fornite dalle tavole Mpeg-PSI. Per i sistemi che adottano lo standard DVB, i contenuti di questa informazione addizionale e la sua sintassi sono definiti dalla specifica DVB-Service Information (DVB-SI). La DVB-SI e la Mpeg-PSI costituiscono quella che viene chiamata **Service Information (SI)**.

Come già abbiamo avuto modo di vedere precedentemente la PSI fornisce informazioni che permettono la configurazione automatica del ricevitore ed il demultiplexaggio e decodifica di qualsiasi programma/evento contenuto nel TS. La DVB-SI d'altra parte, ha il compito di fornire informazioni su: i

servizi disponibili, gli eventi di ogni servizio, descrizioni testuali e tecniche di qualsiasi elemento (network, servizio, evento, stream...), raggruppamento di eventi in differenti categorie, etc...

Se ricordiamo, le tavole PSI forniscono informazioni solo sul TS nel quale sono contenute, al contrario le tavole DVB-SI possono fornire anche informazioni sui servizi e sugli eventi trasportati da altri TSs, ed addirittura su TSs trasmessi da altri network. Ciò permette la commutazione dell'IRD fra differenti TS in modo impercettibile dall'utente.

La DVB-SI è imprescindibile per:

- La sintonizzazione automatica del IRD a seconda del servizio selezionato.
- La localizzazione dei programmi.
- La Application Programming Interface (API): è il sistema che fornisce connessione fra le applicazioni software (p.e.: EPG) e l'hardware. Facendo un parallelo con un PC, sarebbe il Sistema Operativo dell'IRD.
- La Guida dei Programmi Elettronica (EPG): è l'equivalente alla sezione TV di una rivista o di un giornale. È una applicazione software, creata dai providers di servizi, con l'obiettivo di presentare in modo gradevole e intuitivo tutti i servizi disponibili e agevolare in questo modo la scelta del telespettatore. Questa applicazione normalmente si presenta sotto forma di un menù formato da testo, immagini e persino video clips.
- Accesso Condizionato (CA).

Queste funzionalità sono molto sensibili ad ogni piccolo errore nei dati SI e può negare l'accesso ad un servizio criptato o la non presentazione da parte della EPG di qualche servizio disponibile. È per questa ragione che è vitale che i dati SI non contengano errori.

La sintassi DVB-SI è molto vincolata a quella della PSI. Concretamente è strutturata in nuove tavole che sono trasportate attraverso la struttura Private Section definita dall'Mpeg-PSI.

Le nuove tavole sono:

Bouquet Association Table (BAT)

La BAT fornisce informazioni che riguardano i bouquets. Oltre che indicare il nome del bouquet, fornisce anche lista dei servizi che compongono ogni bouquet.

Service Description Table (SDT)

La SDT contiene dati che descrivono i servizi nel sistema, per esempio: nome dei servizi, il provider di servizi, etc...

Event Information Table (EIT)

La EIT contiene dati concernenti gli eventi o programmi, come: il nome dell'evento, l'ora di inizio, la durata, etc...

A seconda del tipo di evento e del tipo di evento si potrà trasmettere un tipo d'informazione o un altro, grazie a l'utilizzo di descriptors differenti.

Running Status Table (RST)

La RST fornisce il flag di un evento, se si sta trasmettendo o no (running/not running). La RST aggiorna questa informazione e permette la commutazione automatica dell'IRD se questo era stata preventivamente programmato per mostrare un determinato evento.

Time and Date Table (TDT)

La TDT fornisce informazioni relative all'orario corrente nel fuso orario di riferimento, il meridiano di Greenwich.

Time Offset Table (TOT)

La TOT fornisce informazioni relative all'orario e data corrente della zona dove è ricevuto il TS.

Stuffing Table (ST)

La ST si utilizza per validare/invalidare sezioni esistenti.

Selection Information Table (SIT)

La SIT si usa soltanto nei bitstreams finiti (parziali) (per esempio, immagazzinati o incisi). Trasporta un sommario dell'informazione SI richiesta per descrivere gli streams all'interno dei bitstreams finiti (parziali).

Discontinuity Information Table (DIT)

La DIT si usa soltanto nei bitstreams finiti. Si inserisce dove l'informazione SI del bitstream può essere discontinua.

Oltre a definire queste nuove tavole, DVB-SI stabilisce anche i contenuti e la sintassi della **NIT**, che come abbiamo già visto, è una tavola definita dall'Mpeg-2 Systems, ma il suo contenuto rimaneva a disposizione di dell'utente.

Le sezioni DVB-SI, così come la Private Section, sono costituite da alcuni campi fissi e di differenti loops, dove si possono includere differenti descriptors. Questi descriptors come già spiegato sono strutture sintattiche che permettono di trasmettere informazioni addizionale in modo standardizzato. DVB definisce i propri descriptors e in quali tavole può essere utilizzato ciascuno di essi. In questo studio spiegheremo la funzione di questi descriptors e in quali tavole si possono trovare.

Nei capitoli che seguono vedremo come si strutturano e come vengono trasportate le tavole DVB-SI nella sintassi Mpeg-2, definiremo la funzione ed i contenuti di ogni tavola una a una, ed infine per facilitare la comprensione dell'utilizzo delle tavole SI, spiegheremo dal punto di vista della EPG, quale è il processo per localizzare il programma selezionato.

Prima di tutto però, occorrerà chiarire quale è la terminologia che utilizza il DVB per riferirsi a programmi, canali, etc.

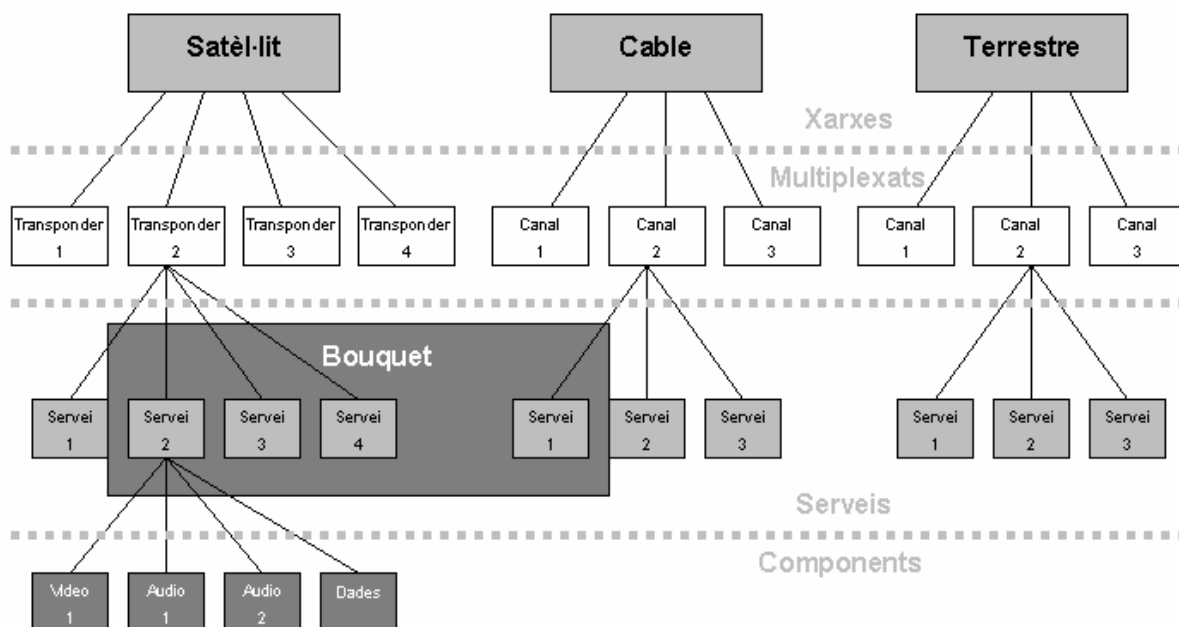
3.2.1. Concetti e terminologia del DVB

Abbiamo già visto che un programma è una collezione di elementi di programma con una base di tempo comune, dove gli elementi di programma possono essere elementary streams o qualsiasi altro stream di dati. La definizione dei programmi si esegue attraverso le Tavole PSI e grazie a ciò si riesce a creare canali virtuali all'interno di un TS.

Questa definizione di "programma" non corrisponde totalmente a quella tradizionalmente utilizzata nella televisione analogica, dove un programma veniva ad essere una collezione di ESs con una base di tempo comune e con un tempo di inizio e di fine comune. Questa última definizione è quella che il DVB denomina evento (= programma convenzionale).

Una sèrie di eventi possono essere trasmessi sequenzialmente in un TS con lo stesso program_number (assegnato nella PAT) per creare quello che tradizionalmente conosciamo come canale di televisione, e che nella terminologia dell'Mpeg-2 e del DVB si chiama servizio (service).

A causa della grande quantità di programmi che può offrire una piattaforma di TV digitale, il DVB introduce un nuovo termine, il bouquet. Un bouquet è raggruppamento di servizi che hanno una tematica comune (sport, notizie, films, ...).



Imatge 3-1

Come già detto un programma si identifica attraverso un program_number che ha significato soltanto all'interno di un TS. Il program_number è un numero intero senza segno di 16 bits, pertanto possono esistere fino a 65535 programmi unici all'interno di un TS (il program_number 0 è riservato per la NIT).

Quando sono disponibili più TSs in un decodificatore (per esempio in un network via cavo), per demultiplexare un programma con successo, il decodificatore deve sapere sia il transport_stream_id che il program_number all'interno del servizio desiderato.

3.3. Trasporto e Struttura delle tavole DVB-SI

Le Tavole DVB-SI sono trasportate nelle Sezioni Privati definite dall'Mpeg-2 Systems. Nel punto 2.3.11 abbiamo spiegato qual'è il meccanismo di funzionamento di queste sezioni.

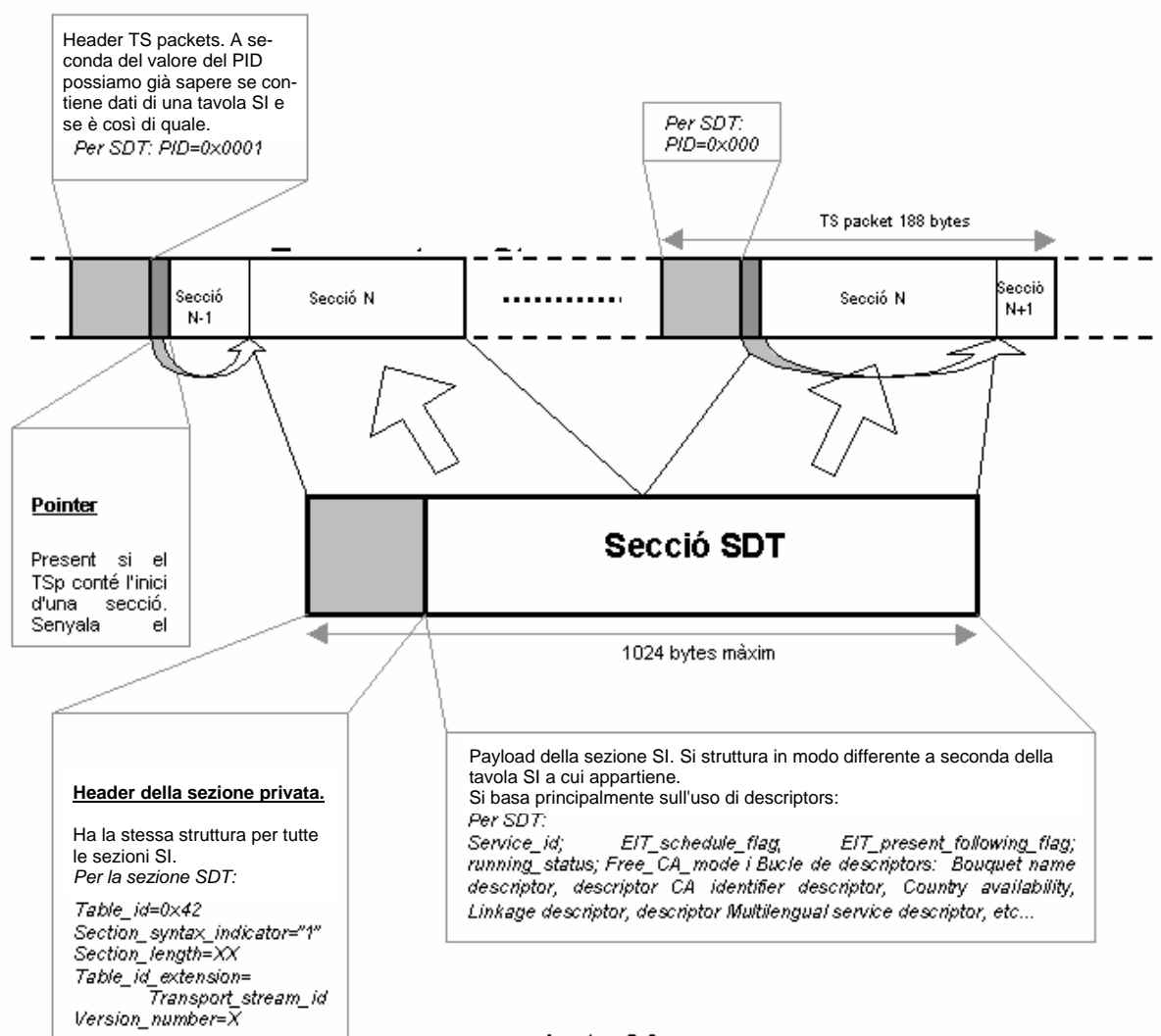
Le tavole NIT, BAT, SDT ed EIT, sono formate da sezioni che mantengono la struttura comune delle Tavole PSI (section_syntax_indicator=1).

Il resto delle tavole non mantengono questa struttura (section_syntax_indicator=0).

Allo stesso modo che le tavole PSI, ogni tavola DVB-SI è formata da varie sotto-tavole. Ogni sotto-tavola è un insieme di sezioni che informano su di uno stesso elemento (network, transport stream, servizio, etc...) a seconda del tipo di sotto-tavola.

Le sezioni sono mappate nei TSps nel modo che abbiamo spiegato nel punto 2.3.5. Possono avere una lunghezza massima di 1024 bytes, salvo le sezioni della tavola EIT, che possono arrivare fino a 4096 bytes.

L'immagine 3-2 mostra in modo schematico il processo di mappatura.



Imatge 3-2

L'intestazione della sezione indica a quale tavola appartiene la sezione, quale tipo di sezione è, quale numero di sezione possiede nella tavola e qual'è versione.

Le informazioni su servizi, eventi, network, ecc. risiedono nel payload della sezione. La sua struttura è differente a seconda della tavola alla quale appartiene l'informazione, però quasi tutte le sezioni si

basano sull'utilizzo di descriptors. I descriptors sono piccole strutture sintattiche definite dal DVB-SI che permettono la descrizione di eventi, servizi, network, ecc.... in modo standardizzato.

Nel capitolo seguente, spiegheremo la funzione e la struttura di ogni tavola DVB-SI, quali descriptors si possono usar all'interno di ognuna, e quale è la sua funzione.

Della sintassi delle tavole DVB indicheremo soltanto il significato preciso di quei campi che siano propri di ognuna delle tavole, e non entreremo nella sintassi ed i campi dei descriptors poichè seria troppo esteso.

Si si desiderano ulteriori informazioni possono essere reperite nel punto 6 dello standard ETSI EN 300 468.

3.4. Tavole DVB-SI

3.4.1. Network Information Table

La NIT fornisce informazioni sui TSs trasmessi da un determinato network e sulle caratteristiche di questo stesso network.

Si utilizza durante i processi di inizializzazione dell'IRD in quanto contiene informazioni sulla sintonizzazione. Questa può essere immagazzinata in una memoria non volatile per minimizzare il tempo d'accesso.

La sua trasmissione è obbligatoria per il network attuale, però si possono anche trasmettere NIT's che informano su altri networks e/o sui TS in essi trasportati. Ciò permette all'IRD di disporre dell'informazione sufficiente per poter commutare su altri networks e ricevere altri TS's.

Ogni sotto-tavola della NIT informa su un solo network. La sotto-tavola che informa sul network attuale elenca obbligatoriamente tutti i TSs presenti in essa.

Ogni sotto-tavola NIT è segmentata in network_information_sections che seguono la struttura seguente:

Tavola 3.3.1-1

| Sintassi | Numero di bits |
|--------------------------------------|---|
| Network_information_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| network_id | 16 |
| reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| reserved_future_use | 4 |
| network_descriptors_length | 12 |
| for(i=0;i<N;i++) | <u>loop descriptors di Network</u> |
| { | |
| descriptor() | |
| } | |
| reserved_future_use | 4 |
| transport_stream_loop_length | 12 |
| for(i=0;i<N;i++) | <u>loop di TSs</u> |
| { | |
| transport_stream_id | 16 |
| originale_network_id | 16 |
| reserved_future_use | 4 |
| transport_descriptors_length | 12 |
| for(j=0;j<N;j++) | <u>loop descriptor di TS</u> |
| { | |
| descriptor() | |
| } | |
| } | |
| CRC_32 | 32 |
| } | |

Se studiamo la tavola sopra riportata vedremo che la sezione comincia con l'intestazione propria della Mpeg-2 Private Section dove il table_id_extension prende il nome di network_id, ed indica su quale network informa questa sezione NIT. Di seguito c'è un primo loop di descrittori che contengono informa-

zioni sul network (descriptors di network). Nel secondo loop si elencano TSs presenti nel network, i per ciascuno di essi vi è un terzo loop di descriptors che informano su di esso (descriptors di TS).

È fondamentale comprendere la funzione dei campi `network_id` ed `original_network_id`.

L'`original_network_id` è stato pensato per identificare univocamente un servizio contenuto in un TS, incluso se questo TS è stato trasferito a un'altro network da quello in cui è stato originato. Un TS può essere univocamente identificato attraverso l'`original_network_id` ed il `transport_stream_id`. Ed un servizio può essere univocamente identificato attraverso l'`original_network_id`, il `transport_stream_id` ed il `service_id`. Perciò ogni `service_id` deve essere unico all'interno di ogni `original_network_id`. Il `network_id` indica dentro di quale network si trasportano i TSs o servizi associati ad esso. Quando un servizio, contenuto in un TS, è trasferito ad un altro network, cambia soltanto il `network_id`, mentre il `original_network_id` resta inalterato.

3.4.2. Descriptors della Network Information Table (NIT)

- Primo loop: Descriptor di Network

In questo loop possono presentarsi i seguenti descriptors:

Linkage Descriptor

Questo descriptor mette in collegamento (link) con un servizio (o TS) di informazioni che può essere visualizzato se il telespettatore richiede informazioni aggiuntive relative a qualcuna delle entità descritte dal Sistema SI. La localizzazione di questo descriptor indica per quale tipi di entità si fornisce questa informazione aggiuntiva.

In questo caso, quando è nel primo loop della NIT, collega ad un servizio che fornisce informazioni sull'operatore di network. Per esempio potrebbe collegarsi con il "Canale di informazione di Cable de Barcelona" o con il "Cable de Barcelona TXT", o con tutti e due, poiché può presentarsi anche più di una volta in questo loop. La sua trasmissione è opzionale.

Il significato del descriptor dipende dal valore del campo `linkage_type`:

- 0x01 collega con un servizio che contiene informazioni sulla network.
- 0x02 collega con un servizio della EPG di informazione sulla network. L'IRD potrà fare uso di questo tipo di link soltanto se è in grado di decodificare il servizio EPG.
- 0x04 collega con un TS che trasporta un gran volume d'informazioni SI. Il TS collegato include come minimo tutta l'informazione SI disponibile a tutti gli altri TSs del network.

Per identificare il servizio connesso fornisce i tre campi necessari: `transport_stream_id`, `original_network_id` e `service_id`.

Multilingual network name descriptor

Questo descriptor si utilizza per trasmettere il nome del network in una o più lingue. Può essere incluso soltanto una volta nel loop descriptor e la sua inclusione è opzionale.

Network name descriptor

Questo descriptor è usato per trasmettere il nome del network, per esempio: ASTRA, EUTELSAT,..... Questo descriptor deve essere usato esattamente una volta in qualsiasi sotto-tavola NIT.

- Terzo Loop: Descriptor dei Transport Streams

In questo loop possono presentarsi i seguenti descriptors:

Delivery system descriptors

I Delivery System Descriptors sono tre descriptors: il *satellite delivery system descriptor*, il *cable delivery system descriptor* ed il *terrestrial delivery system descriptor*.

A seconda del sistema di trasmissione (Cable, Satellite o Terrestre) si usa un descriptor o un altro. La sua funzione è fornire i parametri fisici di ogni TS della network.

- Cavo: frequenza, modulazione, symbol rate, sistemi di correzione d'errore...
- Satellite: frequenza, posizione orbitale, polarizzazione, modulazione, symbol rate....
- Terrestre: frequenza centrale, ampiezza di banda, costellazione, intervallo di guardia, modalità di trasmissione (2K, 8K)....

Tutti e tre hanno una lunghezza di 13 bytes. Questo per facilitarne lo scambio quando un TS viene cambiato di network di distribuzione.

Deve esserci soltanto un `delivery_system_descriptor` ad ogni loop.

Service list descriptor

Questo descriptor è usato per elencare i servizi ed i tipi di servizio di ogni TS. I servizi elencati sono identificati tramite il `service_id` che coincide con il `program_number` assegnato nelle tavole PSI. Il `transport_stream_id` ed l'`original_network_id` necessari per completare l'identificazione univoca di ogni servizio sono forniti all'inizio del loop descriptor della NIT come si può vedere nella tavola precedente.

È ammesso un solo service list descriptor ad ogni loop. La sua trasmissione è opzionale, ma se è presente allora l'elenco dei servizi deve essere completo.

Frequency list descriptor

Questo descriptor fornisce l'elenco delle frequenze addizionali usate per un cert TS che è trasmesso da più frequenze.

È ammesso un solo un frequency list descriptor per ogni loop nel quale ci sia un `delivery_system_descriptor`.

La sua trasmissione è opzionale, però se è presente allora l'elenco deve essere completo.

3.4.3. Bouquet Association Table (BAT)

La BAT fornisce raggruppamenti di servizi, chiamati bouquets, che servono come base per la presentazione ordinata dei servizi all'utente. Il criteri di raggruppamento e le caratteristiche di ogni bouquet sono totalmente definibili dal provider dei servizi, per esempio: film d'azione, serie di telefilm, documentari sugli animali,...

Ogni sotto-tavola BAT è un insieme di sezioni BAT che definiscono interamente un solo bouquet, cioè, listano tutti i servizi che appartengono a questo bouquet. Un bouquet può raggruppare servizi di più di un TS, i quali possono essere anche trasportati in un network differente. Ogni servizio può essere incluso in più di un bouquet.

Poiché la BAT è stata pensata in modo che l'IRD offrisse una gestione basata sui bouquets, occorre assicurare che tutto il servizio sia listato in uno o più bouquets. Se non fosse così, alcuni servizi Benché disponibili non saranno mostrati da questa sistema di gestione e il telespettatore potrà pensare che non siano disponibili. Opzionalmente, per facilitare l'accesso dell'IRD all'informazione dei servizi raggruppati in bouquets, tutti i servizi per cui la BAT fornisce i dati di riferimento devono essere elencati nella Service Description Table (SDT) e si deve fornire informazione NIT di tutti i TSs che contengono qualcuno dei servizi elencati.

La trasmissione della BAT è opzionale.

Ogni sotto-tavola BAT è segmentata in bouquet_association_sections che seguono la struttura seguente:

Tavola 3.3.2-1

| Sintassi | Numero di bits |
|--------------------------------------|---|
| Bouquet_association_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| bouquet_id | 16 |
| reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| reserved_future_use | 4 |
| bouquet_descriptors_length | 12 |
| for(i=0;i<N;i++) | <u>loop di descriptors del bouquet</u> |
| { | |
| descriptor() | |
| } | |
| reserved_future_use | 4 |
| transport_stream_loop_length | 12 |
| for(i=0;i<N;i++) | <u>loop di TSs</u> |
| { | |
| transport_stream_id | 16 |
| original_network_id | 16 |
| reserved_future_use | 4 |
| transport_descriptors_length | 12 |
| for(j=0;j<N;j++) | <u>loop di descriptor dei TS</u> |
| { | |
| descriptor() | |
| } | |
| } | |
| CRC_32 | 32 |
| } | |

Nella tavola soprariportata si può osservare che la BAT e la NIT hanno la stessa struttura. In questo caso il campo `table_id_extension` proprio della intestazione Private Section prende il nome di `bouquet_id` e serve da etichetta identificatrice del bouquet. Il primo loop di descriptors si usa per descrivere il bouquet (nome del bouquet, in quale paese è disponibile...). Nel secondo loop si elencano i TSs che contengono i servizi che il bouquet comprende e nel terzo loop di descriptors vengono elencati questi servizi e di quale tipo sono.

Per esempio, per un bouquet di sport, il loop dei TSs elencherebbe tutti i TSs che contengono sport, ed il loop di descriptors dei TS elencherebbe tutti i servizi di sport contenuti in ogni stream. Il telespettatore selezionerebbe il bouquet dello sport, vedrebbe la lista dei servizi disponibili e sceglierebbe il servizio/evento desiderato.

3.4.4. Descriptors della Bouquet Association Table (BAT)

- Primo Loop: Descriptors del Bouquet

In questo loop possono essere inclusi i seguenti descriptors:

Bouquet name descriptor

Questo descriptor si usa per trasmettere il nome (sotto forma di testo) del bouquet, per esempio: "el Bouquet de Notícias", "el Bouquet de Pel·lícules",... Questo descriptor appare una sola volta in ogni sotto-tavola BAT, poichè come abbiamo già detto ogni sotto-tavola BAT definisce interamente un solo bouquet.

CA identifier descriptor

Questo descriptor indica se un bouquet, un servizio o un evento è associato ad un sistema di accesso condizionato ed identifica il tipo di sistema. Se lo troviamo nella BAT fa sempre riferimento ad un bouquet.

La sua presenza è ammessa una sola volta all'interno di questo loop e la sua trasmissione è opzionale.

Country availability descriptor

Questo descriptor indica in quali paesi è disponibile il bouquet associato e in quali non lo è. Il concetto di 'disponibile' non ha niente a che vedere con 'trasmesso', la trasmissione di un bouquet arriva comunque ad un determinato paese per il quale il bouquet può essere definito come non disponibile. Questa funzionalità non ha nessuna relazione con i sistemi d'Accesso Condizionato.

Sarà opportuno che gli IRD's facciano uso di questo descriptor in modo da non visualizzare i bouquets che, Benché vengano ricevuti, non sono disponibili, per evitare la frustrazione degli utenti.

In ogni sotto-tavola BAT può presentarsi al massimo due volte. Una per indicare la lista dei paesi dove il bouquet è disponibile ed un'altra per indicare la lista di paesi dove il bouquet non è disponibile. Se appare soltanto una volta per indicare i paesi dove il servizio è disponibile (non è disponibile), si interpreta che per qualsiasi altro paese non è disponibile (è disponibile).

La sua trasmissione è opzionale.

Linkage Descriptor

(Vedere il primo paragrafo `Linkage_descriptor` nel punto 3.4.2)

Se appare nel primo loop della BAT collega ad un servizio che fornisce informazioni sul bouquet. È ammessa la sua presenza nel loop più di una volta. Potrebbe per esempio collegarsi al "Grans Pel·lícules" e al "Grans Pel·lícules TXT".

La sua trasmissione è opzionale.

Il significato del descriptor dipende del valore del campo linkage_type:

- 0x01 collega con un servizio che contiene informazioni sul bouquet.
- 0x02 collega con un servizio della EPG di informazione sul bouquet. L'IRD potrà fare uso di questo tipo di link soltanto se è in grado di decodificare il servizio EPG.
- 0x04 collega con un TS che trasporta un gran volume d'informazioni SI. Il TS collegato include come minimo tutta l'informazione SI disponibile a tutti gli altri TSs del bouquet.

Multilingual bouquet name descriptor

Questo descriptor si utilizza per trasmettere il nome del bouquet in una o più lingue. Può essere incluso soltanto una volta nel loop descriptor e la sua inclusione è opzionale.

- Terzo Loop: Descriptors dei Transport Streams

Ci si ricordi che i contenuti di questo loop sono vincolati sempre ad un solo Transport Stream. Al suo interno vi possono apparire soltanto i seguenti descriptors:

Service list descriptor

Questo descriptor si usa per listare i servizi ed i tipi di servizio di ogni TS associato al bouquet in questione. Permette all'IRD di trovare tutti i servizi associati ad un bouquet specifico.

Deve essere presente in tutta BAT ed è ammesso una sola volta per ogni loop

3.4.5. Service Description Table (SDT) information

La SDT si usa per fornire l'elenco dei nomi e di altri parametri dei servizi inclusi nei TSs.

Ogni sotto-tavola SDT informa su un solo TS. La sua trasmissione è obbligatoria per ogni TS e deve elencare come minimo tutti i servizi di questo TS. Le sotto-tavole SDT possono informare anche su altri TSs diversi da quello attuale, se è così devono avere il campo `table_id=0x46`, e se informano sul TS attuale il `table_id` deve essere uguale a `0x42`.

Si raccomanda che una volta assegnato un `service_id` ad un servizio specifico all'interno di un network, questo `service_id` venga poi mantenuto senza cambiamenti per permettere agli IRDs di implementare funzioni come la lista dei canali preferiti.

Ogni sotto-tavola SDT viene segmentata in `service_description_sections` che seguono la struttura seguente:

Tavola 3.3.3-1

| Sintassi | Numero di bits |
|--------------------------------------|---|
| Service_description_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| transport_stream_id | 16 |
| reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| original_network_id | 16 |
| reserved_future_use | 4 |
| for(i=0;i<N;i++) | <u>loop dei servizi</u> |
| { | |
| service_id | 16 |
| reserved_future_use | 6 |
| EIT_schedule_flag | 1 |
| EIT_present_following_flag | 1 |
| Running_status | 3 |
| Free_CA_mode | 1 |
| Descriptors_loop_length | 12 |
| for(i=0;i<N;i++) | <u>loop dei descriptors del servizio</u> |
| { | |
| descriptor() | |
| } | |
| } | |
| CRC_32 | 32 |
| } | |

Possiamo osservare che il campo `table_id_extension` proprio dell'intestazione (header) Private Section prende il nome di `transport_stream_id` il quale serve per indicare su quale TS questa sezione fornisce informazioni. Nel loop dei servizi vengono elencati tutti i servizi del TS. Per ognuno si indica se vi è informazione EIT disponibile, quale è il suo stato, se qualcuno degli streams che compongono il servizio è criptato. Si dispone di un loop di descriptors per ogni servizio.

3.4.6. Descriptors della Service Description Table (SDT)

Bouquet name descriptor

Questo descriptor si usa per trasmettere il nome del bouquet o dei bouquets ai quali è associato il servizio, per esempio "Il Bouquet de Noticias" e/o "Canals de Pel·lícules".

La sua trasmissione nella SDT è opzionale ed in realtà viene ad essere uno 'spreco' dell'ampiezza di banda, poiché la stessa informazione può essere trasmessa con maggiore efficienza attraverso la BAT.

CA identifier descriptor

(Vedere CA identifier descriptor nel punto 3.4.2.)

In questo caso fa riferimento ad un servizio.

Il CA_identifier_descriptor non è coinvolto in nessuna funzione di controllo di Accesso Condizionato, indica dolo all'IRD se un servizio è sottoposto a CA e di quale sistema CA si tratta. L'IRD può decidere in questo caso se questo servizio è accessibile oppure no, e di conseguenza presentarlo come disponibile oppure no. Ciò evita la frustrazione del telespettatore causata dalla presentazione di servizi che non sono accessibili.

La sua trasmissione è opzionale ed è ammesso una sola volta all'interno del loop.

Country availability descriptor

Questo descriptor ha la stessa funzione descritta nel punto 3.4.4, ma in questo caso fa riferimento ad un servizio invece che ad un bouquet. Le sue condizioni di utilizzo sono anch'esse le stesse.

Data broadcast descriptor

Questo descriptor si usa per identificare servizi/eventi di diffusione di dati in ambiente DVB, e può essere usato anche per fornire una descrizione testuale dei dati.

Se identifica un servizio soltanto si fa uso di questo descriptor nella SDT, al contrario se identifica un evento si usa sia nella SDT che nella EIT.

Il tipo di servizio di dati viene indicato attraverso il campo data_broadcast_id.

Linkage descriptor

(Vedere primo paragrafo Linkage_descriptor nel punto 3.4.2)

In questo caso collega ad un altro servizio che fornisce informazione addizionale sul servizio al quale va associato. Un esempio potrebbe essere la disponibilità di una opzione, da parte dell'IRD, chiamata "informazione sul servizio", che faccia commutare l'IRD su questo servizio collegato. La sua trasmissione è opzionale ed è ammessa anche più di una volta in un loop.

Il suo significato dipende dal valore del campo linkage_type:

- 0x01 collega con un servizio che contiene informazioni sul servizio.
- 0x02 collega con un servizio della EPG di informazione sul servizio. L'IRD potrà fare uso di questo tipo di link soltanto se è in grado di decodificare il servizio EPG.
- 0x03 collega con un servizio di sostituzione del CA. Un esempio di utilizzo per il quale è stato pensato potrebbe essere la commutazione automatica dell'IRD al servizio di sostituzione se il sistema CA nega l'accesso al servizio in questione.
- 0x05 collega con un servizio di sostituzione del servizio in questione. Un esempio potrebbe essere la commutazione automatica dell'IRD al servizio sostitutivo quando il servizio selezionato si trova in stato "not running".

Per identificare il servizio connesso fornisce i tre campi necessari: transport_stream_id, original_network_id e service_id.

Mosaic descriptor

Questo descriptor è trasportato nella SDT e/o nella PMT. Si usa per descrivere servizi di mosaico. Come descritto nel paragrafo 5.2 (???)

Multilingual service descriptor

Questo descriptor si usa per trasmettere il nome del provider di servizi ed il nome del servizio, in una o più lingue. Può essere incluso soltanto una volta nel loop descriptor e la sua inclusione è opzionale.

NVOD reference descriptor

Questo descriptor insieme ai descriptors time shifted service e time shifted event forniscono un meccanismo per descrivere efficientemente i servizi Near Video on Demand (NVOD).

Un servizio NVOD è in realtà un gruppo di servizi uguali, ma sfasati nel tempo l'uno rispetto all'altro, in modo che il telespettatore possa in qualsiasi momento accedere all'evento in un punto più vicino (da qui la parola Near) all'inizio dello stesso. Per ogni servizio NVOD si stabilisce sempre un servizio di riferimento (reference service), questo servizio è un **servizio fittizio** (non corrisponde a nessun program_number nella PSI-PMT) che ha associati i descriptors degli eventi contenuti nel servizio NVOD. Questi descriptors sono comuni a tutti gli eventi "copia", in modo che i servizi che formano il servizio NVOD non devono trasportare i "suoi propri" descriptors per ogni evento, ma si associano a questo servizio di riferimento. Per indicare questa associazione ognuno dei servizi è descritto nella SDT appropriata in modo del tutto normale, ma facendo uso di un Time shifted service descriptor (vedere in questo stesso punto: Time shifted service descriptor) mentre ogni evento è descritto nella EIT facendo uso di un Time shifted event descriptor per associarlo ad un evento concreto nel servizio di riferimento.

Con tutto ciò si evita di inviare informazione ridondante.

Il NVOD reference descriptor è legato a questo servizio di riferimento. La sua funzione è quella di elencare tutti i servizi che insieme formano un servizio NVOD e si identifica attraverso il transport_stream_id, original_network_id e service_id.

Service descriptor

Questo descriptor contiene le identificazioni testuali di base di un servizio come il nome ed il nome del provider. La sua trasmissione è obbligatoria e la sua presenza è ammessa una sola volta nel loop.

Gli IRD's lo utilizzano per presentare all'utente i nomi dei servizi.

Telephone descriptor

Questo descriptor si usa per indicare un numero di telefono, che può essere utilizzato da un eventuale modem per l'implementazione canali o funzioni interattive.

La sua trasmissione è opzionale e può apparire più di una volta nel loop.

Time shifted service descriptor

Questo descriptor si usa al posto della maggioranza dei descriptors di servizio precedentemente descritti, nei servizi che sono una copia sfasata nel tempo di un altro servizio. La sua funzione è fornire un collegamento (link) ad un servizio di riferimento NVOD che fa riferimento a tutte le descrizioni necessarie e comuni a tutti i servizi che sono copie sfasate di uno stesso servizio.

È per questa ragione che la maggioranza dei descriptors non possono essere presenti quando vi è un Time shifted service descriptor. (En questo lavoro non abbiamo voluto entrar in piccoli dettagli e non abbiamo indicato quali sono questi descriptors).

È obbligatoria la sua trasmissione nei servizi elencati in un NVOD_reference_descriptor, ed è ammesso soltanto una volta nel loop.

3.4.7. Event Information Table (EIT) information

La EIT si usa per trasmettere le informazioni relative agli eventi contenuti da ogni servizio. Ogni servizio ha associata una sotto-tavola EIT indipendente.

A seconda del tipo di informazione trasportata si definiscono due tipi di sezioni: la EIT Present/Following e la EIT Schedule.

La EIT Present/Following contiene soltanto informazione sull'evento attuale o sull'evento successivo (cronologicamente parlando) di un servizio trasportato nel TS attuale o in un altro TS. Le informazioni di base trasportate da una sezione EIT P/F sono il tempo di inizio, la durata ed il flag (running status) di uno di questi due eventi. Se fornisce informazioni su un servizio del TS attuale viene identificato con il campo `table_id="0x4E"`, se no viene identificato con il `table_id="0x04F"`.

La EIT Schedule fornisce un elenco di eventi ordinati cronologicamente in formato di programmazione o d'orario. In questo caso l'elenco può arrivare fino ad a eventi molto in là nel tempo rispetto all'evento successivo, da giorni fino anche a settimane o mesi. La sua trasmissione è opzionale e si identifica con il campo `table_id="0x50" fino a 0x5F"` per i servizi del TS attuale, e con il `table_id="0x60" fino a 0x6F"` per i servizi di altri TSs.

Tutte le sezioni EIT hanno la stessa struttura. Nella seguente tavola si può osservare come La EIT P/F e la EIT S seguano la stessa sintassi.

Tavola 3.3.4-1

| Sintassi | Numero di bits |
|------------------------------------|--|
| Event_information_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| service_id | 16 |
| reserved | 2 |
| version_number | 5 |
| current_next_indicator | 1 |
| section_number | 8 |
| last_section_number | 8 |
| transport_stream_id | 16 |
| original_network_id | 16 |
| segment_last_section_number | 8 |
| last_table_id | 8 |
| for(i=0;i<N;i++) | <u>loop degli eventi</u> |
| { | |
| event_id | 16 |
| start_time | 40 |
| duration | 24 |
| Running_status | 3 |
| Free_CA_mode | 1 |
| Descriptors_loop_length | 12 |
| for(i=0;i<N;i++) | <u>loop dei descriptors dell'evento</u> |
| { | |
| descriptor() | |
| } | |
| } | |
| CRC_32 | 32 |
| } | |

In questo caso il campo `table_id_extension` proprio dello header Private Section prende il nome di `service_id` ed indica a quale servizio sono associati gli eventi descritti in questa sezione. Il `service_id` concorda con il `program_number` della PSI-PMT.

Il campo `segment_last_section_number` serve per raggruppare le sezioni EIT schedule in segmenti, come spiegheremo più avanti. Si non si tratta di una sotto-tavola segmentata il valore di questo campo è uguale al `last_section_number`. Il campo `last_table_id` prende il valore del `table_id` più elevato utilizzato nella tavola EIT.

Il primo loop di servizi si usa per elencare gli eventi e fornire informazioni sugli stessi (istante di inizio, durata,...). Il secondo loop è nidificato nel primo e la sua funzione è permettere la descrizione di ogni evento.

3.4.8. EIT Present/Following information

Ogni servizio ha associate due sezioni EIT P/F, una per l'evento presente, che s'identifica con il `section_number=0x00`, ed un'altra per l'evento successivo, che si identifica con il `section_number=0x01`.

In ogni istante per qualsiasi servizio può esserci soltanto al massimo un solo event presente ed un solo evento successivo. Nel caso di servizi NVOD ciò non è così e pertanto le seguenti spiegazioni non sono applicabili a questo tipo di servizio. Quando non vi sono evento presenti o successivi (per esempio: Fine programmazione) si trasmette una sezione vuota della EIT P/F.

La lunghezza massima di una sezione è di 4096 bytes pertanto la descrizione di un solo evento deve essere inferiore a questa. La descrizione degli eventi è opportuno che avvenga in ordine ascendente secondo l'`event_id`.

Il campo `running_status` va interpretato secondo la tavola 3.3.4-2 per l'evento presente e secondo la tavola 3.3.4-3 per l'evento successivo.

Tavola 3.3.4-2

| | |
|-------------------------|--|
| undefined | La sola informazione trasmessa è lo stato. L'evento si sta trasmettendo in questo istante |
| running | L'evento si sta trasmettendo in questo istante. |
| not running | L'evento è nominalmente presente però in questo istante non lo si sta trasmettendo o perchè è terminato o perchè non è ancora cominciato. |
| pausing | L'evento è nominalmente presente e si è già cominciato a trasmetterlo ma in questo istante quello che si sta trasmettendo non fa parte dell'evento (per esempio annunci commerciali) e la sua trasmissione riprenderà più tardi. |
| starts in a few seconds | Il flag dell'evento tra pochi secondi diventerà "running". La sua utilità è permettere una registrazione programata corretta da parte degli IRDs o dei VCRs. |

Tavola 3.3.4-3

| | |
|-------------------------|--|
| Undefined | La sola informazione trasmessa è lo stato. L'evento è il prossimo ad essere trasmesso. |
| running | Non ammesso. Non ha significato in questo contesto. |
| not running | L'evento, in questo istante, non si sta trasmettendo. |
| pausing | Indica che lo stato dell'evento seguente è stato 'running' per alcuni istanti, però ora ci è un altro evento sovrapposto. Durante tutto il tempo in cui l'evento seguente rimane in pausing, l'event sovrapposto deve essere definito come come l'evento presente. Quando l'evento presente terminerà, tornerà allo stato running. |
| starts in a few seconds | Fra pochi secondi il flag dell'evento diventerà "running". |

Nel campo `duration` si codifica il tempo di durata dell'evento. Questo tempo include anche la durata di tutti gli intervalli in cui l'evento rimane col flag "not running" o "paused".

Nel campo `start_time` si codifica l'istante di inizio dell'evento. Questo istante è l'inizio dell'evento nel suo insieme; per esempio, se vi è una pausa non si deve indicare l'orario di re-inizio dell'evento dopo di questa.

L'istante di inizio di un evento più la sua durata può essere minore dell'istante di inizio dell'evento seguente. Ciò significa che sono permessi intervalli vuoti fra gli eventi.

3.4.9. EIT Schedule Information

La EIT Schedule Information contempla gli eventi che possono aver luogo nelle prossime ore e fino a 64 giorni successivi. Per questa ragione richiede una organizzazione più complessa della EIT P/F.

In un TS si dispone di 16 sotto-tavole per descrivere il TS attuale (`table_id's=0x50-0x5F`) e di altre 16 per il resto dei TSs (`table_id's=0x60-0x6F`). Come già visto in altri paragrafi ogni sotto-tavola è formata da 256 sezioni. Queste 256 sezioni vengono raggruppate in 32 segmenti di 8 sezioni ciascuno. In modo che il segmento #0 contiene le sezioni da 0 a 7, il segmento #1 le sezioni da 8 a 15

I segmenti si definiscono tramite il campo `segment_last_section_number` che prende il `section_number` dell'ultima sezione che forma il segmento.

Ogni segmento contiene le informazioni degli eventi che iniziano entro un periodo di tre ore. Pertanto se consideriamo che abbiamo 16 sotto-tavole, di 32 segmenti ognuna, ogni segmento equivale a 3 ore, per cui al massimo si possono fornire informazioni sui prossimi 64 giorni. Diciamo al massimo perchè la trasmissione della EIT S è opzionale.

Gli eventi sono ordinati cronologicamente nei segments in modo che se non si usano le otto sezioni di un segmento i dati appaiono sempre nelle prime sezioni. In ogni caso, per indicare qual'è l'ultima sezione usata si utilizza il campo `segment_last_section_number`. In questo campo si conserva il numero risultante di:

[il numero della prima sezione del segmento] - [il numero di sezioni con dati] -1.

Tutti gli istanti di inizio degli eventi hanno come riferimento le ore 00:00 Universal Time Coordinated (UTC), in modo che il segmento #0 contiene gli eventi che iniziano fra le ore 00:00 UTC e le ore 02:59:59 UTC, il segmento #1 contiene gli eventi che iniziano fra le ore 03:00 UTC e le ore 05:59:59, e così via....

Per qualsiasi evento il campo `running_status` risulta undefined.

I segmentis che corrispondono ad eventi passati sono sostituiti da segmenti vuoti.

(Le tavole EIT Schedule non sono applicabili agli eventi NVOD Reference Services, poichè hanno orari di inizio indefiniti.)

3.4.10. Descriptors della Event Info Table (EIT)

Component descriptor

Questo descriptor si usa per indicare tutti gli streams vincolati ad un evento. Stabilisce inoltre di quale tipo sono (video 16:9, EBU Teletext sottotitoli,...) e permette una descrizione testuale di ognuno di essi.

La sua trasmissione è obbligatoria nell'EIT present/Following del TS attuale ed è opzionale per altri EIT's. Può apparire più di una volta nel loop in quanto normalmente vi è più di uno stream per ogni evento. È utile per indicare quali streams saranno disponibili per eventi futuri.

Content descriptor

Questo descriptor si usa per classificare il contenuto di un evento. Utilizza due campi, il content_nibble_level_1 ed il content_nibble_level_2. Col primo stabilisce un primo livello di classificazione: Movie/Drama, News/Current affairs, Show/Game, Spots... Col secondo si entra più nel dettaglio, per esempio per Movies abbiamo: detective/thriller, adventure/western/war, science fiction/fantasy/horror...

La sua trasmissione è opzionale ed è ammesso soltanto un content descriptor nel loop descriptor, però c'è la possibilità di trasmettere più di un termine di classificazione in quanto è previsto un loop nidificato nello stesso descriptor.

Data broadcast descriptor

(Vedere Data broadcast descriptor al punto 3.4.6.)

Short event descriptor

Questo descriptor si usa per trasmettere il nome ed una breve descrizione testuale dell'evento. Attraverso il campo ISO_639_language_code indica in quale lingua sono scritti il titolo ed il testo.

La sua trasmissione è obbligatoria e non è ammessa la sua presenza più di una volta nel loop per una stessa lingua.

Extended event descriptor

Questa descrizione si usa per trasmettere una maggior qualità di informazioni testuali su un evento rispetto a quella che permette di fornire il short_event_descriptor. La sua capacità è di 255 bytes, se si vuole trasmettere una quantità di dati superiore si possono usare più di una volta nel loop (ciò è valido anche per dati di una stessa lingua).

Quando si usa più di un extended event descriptor per una stessa lingua, questi si associano attraverso il campo descriptor_number (vale 0x00 ogni volta che si inizia una associazione) ed il last_descriptor_number.

Nel suo interno, i dati inviati possono venir strutturati in due colonne, una per il campo descriptor e l'altra per il testo. Per esempio nella prima colonna ci potrebbe essere "Productor" e nella seconda "John Cushak".

Linkage descriptor

(Vedere primo paragrafo Linkage_descriptor nel punto 3.4.1)

Si appare in questo loop connette con un servizio che contiene informazione sull'evento. In questo caso il campo linkage_type può prendere soltanto il valore 0x01. Un esempio di utilizzo per il quale è stato pensato è che IRD possa includere una funzione denominata "informazione sull'evento" che faccia sintonizzare l'IRD sul servizio connesso.

La sua trasmissione è opzionale e la sua presenza è permessa più di una volta nel loop.

Multilingual component descriptor

Questo descriptor trasmette una descrizione testuale di un componente (stream) dell'evento in una o più lingue. Per identificare il componente si utilizza il campo `component_tag` che prende lo stesso valore del campo con ugual nome che vi è nello Stream Identifier Descriptor. Lo Stream Identifier Descriptor deve essere incluso nella tavola PSI_PMT quando si desidera fare una descrizione testuale di un componente.

La sua inclusione è opzionale e può essere inclusa soltanto una volta nel loop di descriptors per ogni componente (stream) dell'evento.

Parental rating descriptor

Questo descriptor si usa per stabilire una classificazione dei programmi basata sull'età o su qualsiasi altro criterio, per impedire ai bambini la visione di programmi a loro non adatti. La sua trasmissione è opzionale e può apparire soltanto una volta ogni loop.

All'interno del descriptor si possono usare `country_codes` per stabilire gruppi di paesi. Si vi è più di una voce per uno stesso paese, allora la prima voce ha la priorità su quelle successive. Per esempio il seguente caso indica una età minima di 12 anni per il Regno Unito e di 18 per il resto dei paesi.

`Country_code=UK`

`Rating=0x0F`

`Country_code=all countries`

`Rating=0x09`

Telephone descriptor

(Vedere Telephone descriptor nel punto 3.4.6)

Time shifted event descriptor

Questo descriptor si usa negli eventi che sono una copia spostata nel tempo di un altro evento, al posto dei descriptors di evento descritti precedentemente. La sua funzione è fornire un collegamento (link) ad un `reference_event_id` del servizio di riferimento NVOD, che fornisce tutte le descrizioni necessarie per gli eventi che sono copie spostate di questo `reference_event_id`.

Quando è presente in un loop, non è ammesso nessun altro descriptor e la sua trasmissione è obbligatoria nel caso di NVOD, poiché senza di esso l'accesso alla SI (Service Information) degli eventi NVOD non è possibile.

3.4.11. Time and Date Table (TDT)

La TDT è usata per sincronizzare il clock interno dell'IRD e si deve trasmettere come minimo ogni 30 secondi.

La TDT trasmette la data e l'ora attuale (UTC). L'ora è codificata nel formato Modified Julian Date (MJD) (Vedere EN 300 468).

La TDT consiste in una sola sezione con la seguente struttura:

Tavola 3.3.6-1

| Sintassi | Numero di bits |
|----------------------------|----------------|
| time_date_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| UTC_time | 40 |
| } | |

3.4.12. Time Offset Table

La TOT trasmette la data e l'ora attuale (UTC) ed informazioni sulla differenza rispetto al fuso local (time offset). Tutte le informazioni riferentesi al tempo va codificata in formato MJD.

La TOT consiste in una sola sezione conla seguente struttura:

Tavola 3.3.7-1

| Sintassi | Numero di bits |
|------------------------------|----------------|
| time_offset_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator = 0 | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| UTC_time | 40 |
| reserved | 4 |
| descriptors_loop_length | 12 |
| for(i=0;i<N;i++) | |
| { | |
| descriptor() | |
| } | |
| CRC_32 | |
| } | |

Si usa per sincronizzare il clock del IRD con l'orario locale.

La sua trasmissione è opzionale, però se viene trasmessa, occorre farlo come minimo ogni 30 secondi.

3.4.13. Descriptors della Time Offset Table (TOT)

Local time offset descriptor

Questo descriptor si usa per indicare la differenza di orario fra l'ora UTC e l'ora locale e per fare la regolazione automatica dell'IRD quando si passa dall'ora legale a quella solare e viceversa.

I dati forniti dal descriptor normalmente saranno costanti. Cambieranno nelle transmissioni da estate a inverno e viceversa. Questo campo può apparire più di una volta, poichè un paese può abbracciare più di una zona oraria. Il campo country_region_id identifica le differenti regioni del paese fra le differenti zone orarie.

Esempio:

Country_code: x9yy (Gruppo di paesi del Continente Europeo)

Country_region_id '000000'

Local_time_offset_polarity '0'

Local_time_offset '0000000100000000' (Inverno: 1 ora)

'0000001000000000' (Estate: 3 ore)

Questo descriptor è utile per le seguenti applicazioni:

- Visualizzazione dell'ora locale attuale sul display dell'IRD o sul teleschermo del TV.
- Visualizzazione della programmazione rispettando l'ora locale.
- Programmazione del VCR o dell'IRD rispettando l'ora locale.

3.4.14. Running Status Table (RST)

Le Running Status Sections sono utilizzate per aggiornare rapidamente il running status di uno o più eventi. Ciò è necessario quando un evento comincia prima o dopo l'orario previsto, a causa di cambiamenti di programmazione. A differenza di altre tavole SI che normalmente vengono trasmesse ripetitivamente, le sezioni RST si inviano soltanto una volta, nell'istante l'istante in cui lo stato dell'evento cambia. La trasmissione di una RST invalida lo stato dell'evento in questione trasmesso precedentemente dalla EIT Present/Following, ma non esiste nessun meccanismo per aggiornare le RST's. La seguente volta che la EIT viene trasmessa deve contenere il running status aggiornato.

La funzione di questo meccanismo opzionale è permettere agli IRD's o VCR's una commutazione estremamente accurata all'inizio dell'evento.

La RST è segmentata in sezioni che seguono la struttura seguente:

Tavola 3.3.8-1

| Sintassi | Numero di bits |
|---------------------------------|----------------|
| Running_status_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| for (i=0;i<N;i++) | |
| { | |
| transport_stream_id | 16 |
| original_network_id | 16 |
| service_id | 16 |
| event_id | 16 |
| reserved_future_use | 5 |
| running_status | 3 |
| } | |
| } | |

3.4.15.- Stuffing Table (ST)

Una sezione di stuffing può aver luogo ovunque si permessa una sezione SI. Le ST sono usate per sostituire o invalidare sia le sotto-tavole che le tavole SI. Per garantire coerenza non si può sostituire o invalidar soltanto determinate sezioni di una sotto-tavola ma devono essere sostituite o invalidate tutte le sezioni di una sotto-tavola. Così quando una sezione di una sotto-tavola è sovrascritta, allora tutte le sezioni di questa sotto-tavola devono essere anch'esse sovrascritte (stuffed) in modo da mantenere l'integrità del campo section_number.

Tavola 3.3.9-1

| Sintassi | Numero di bits |
|-----------------------------|----------------|
| Stuffing_section() | |
| { | |
| table_id | 8 |
| section_syntax_indicator | 1 |
| reserved_future_use | 1 |
| reserved | 2 |
| section_length | 12 |
| for (i=0;i<N;i++) | |
| { | |
| data_byte | 8 |
| } | |
| } | |

Tramite il valore del table_id si sceglie la sotto-tavola da invalidare.

3.4.16. *Altres descriptors*

Teletext descriptor

Si usa nella PMT per identificare i dati di Teletext EBU che sono codificati secondo la norma EN 300 472. Questo descriptor è usato una volta nel campo ES_info appropriato per qualsiasi stream che contenga dati Teletext EBU. Il descriptor permette l'identificazione della lingua ed il tipo di pagina: sottitoli, pagina iniziale, ...

Stuffing descriptor

Questo descriptor è ammesso ovunque nella SI dove è permessa la presenza di descriptors. Si usa per riempire tavole per qualsiasi ragione o per disabilitare descriptors che non son più validi (per esempio nel caso di remultiplexing). Gli IRD's devono omettere questo descriptor.

Descriptors sconosciuti

Quando l'IRD trova un descriptor definito per DVB in un punto in cui non è previsto, o un descriptor con un identificatore non riconosciuto, lo ometterà facendo uso del campo che indica la lunghezza e continuerà l'elaborazione degli altri dati ignorando i dati 'strani'.

3.4.17. Sommario della localizzazione delle Sezioni e dei Descriptors

Per identificare i TSps che contengono sezioni delle Tavole SI E per sapere a quale tavola appartiene ogni sezione, il DVB-SI stabilisce le seguenti relazioni.

Tavola 3.3.11-1

| Tavola | PID |
|-------------------------|------------------|
| PAT | 0x0000 |
| CAT | 0x0001 |
| TSDT | 0x0002 |
| Reserved | 0x0003 to 0x000F |
| NIT, ST | 0x0010 |
| SDT, BAT, ST | 0x0011 |
| EIT, ST | 0x0012 |
| RST, ST | 0x0013 |
| TDT, TUTTO, ST | 0x0014 |
| Network synchronization | 0x0015 |
| Reserved for future use | 0x0016 to 0x001D |
| DIT | 0x001E |
| SIT | 0x001F |

Il PID ci indica a quale tipi di tavola appartiene il pacchetto TS. Ricordiamo che l'Mpeg-2 Systems lasciava i valori di PID fra 0x0010 e 0x1FFE come definibili dall'utente.

Tavola 3.3.11-2

| Valore Table_id | Description |
|-----------------|--|
| 0x00 | Program_association_section |
| 0x01 | Program_map_section |
| 0x02 | Conditional_access_section |
| 0x03 | Transport_stream_description_section |
| 0x04 to 0x3F | Reserved |
| 0x40 | Network_information_section - actual_network |
| 0x41 | Network_information_section - other_network |
| 0x42 | Service_description_section - actual_transport_stream |
| 0x43 to 0x45 | Reserved for future use |
| 0x46 | Service_description_section - other_transport_stream |
| 0x47 to 0x49 | Reserved for future use |
| 0x4A | Bouquet_association_section |
| 0x4B to 0x4D | Reserved for future use |
| 0x4E | Event_information_section - actual_transport_stream, present/following |
| 0x4F | event_information_section - other_transport_stream, present/following |
| 0x50 to 0x5F | event_information_section - actual_transport_stream, schedule |
| 0x60 to 0x6F | event_information_section - other_transport_stream, schedule |
| 0x70 | time_date_section |
| 0x71 | running_status_section |

| | |
|--------------|-----------------------------------|
| 0x72 | stuffing_section |
| 0x73 | time_offset_section |
| 0x74 to 0x7D | reserved for future use |
| 0x7E | discontinuity_information_section |
| 0x7F | selection_information_section |
| 0x80 to 0xFE | user defined |
| 0xFF | Reserved |

Il Table_id ci indica di quale tipo di sezione SI si tratta. Ricordiamo che l'Mpeg-2 Systems lasciava i valori di Table_id fra 0x40 e 0xFE, come definibili dall'utente.

A livello di trasporto, DVB-SI esige un tempo minimo e massimo fra l'ultimo byte di una sezione ed il primo byte della seguente sezione trasmessa con lo stesso PID, table_id, table_id_extension. Nella tavola qui di seguito si riassumono questi tempi.

Tavola 3.3.11-3

| Sezione | Tempo minimo fra sezioni | Tempo massimo fra sezioni |
|---------|--------------------------|---------------------------|
| NIT | 25 ms | 10 seg |
| SDT | 25 ms | 2 seg* |
| BAT | 25 ms | 10 seg |
| EIT | 25 ms | 2 seg* |
| RST | 25 ms | - |
| TDT | 25 ms | 30 seg |
| TOT | 25 ms | 30 seg |
| ST | 25 ms | - |
| PAT | 25 ms | 0,1 seg |
| CAT | 25 ms | 0,1 seg |
| PMT | 25 ms | 0,1 seg |

* Nota :È il tempo per il Transport Stream attuale (ETR-211 Cap 4.4)

Infine presentiamo una tavola dove è indicato quali tavole sono obbligatorie e quali opzionali

Tavola 3.3.11-4

| Tavole Obbligatorie | Tavole Opzionali |
|--|---|
| NIT per il TS attuale (PID 0x0010) | NIT per un altro TS (PID 0x0010) |
| SDT per il TS attuale (PID 0x0011) | SDT per un altro TS (PID 0x0011) |
| EIT present/following per il TS attuale (PID 0x0012) | EIT schedule per il TS attuale, e present/following e schedule per un altro TS (PID 0x0012) |
| TDT per qualsiasi TS (PID 0x0014) | TOT per qualsiasi TS (PID 0x0014) |
| | BAT per qualsiasi TS (PID 0x0011) |
| | RST per qualsiasi TS (PID 0x0013) |
| | BAT per qualsiasi TS (da PID 0x0010 a 0x0014) |

Nota: "per un altro TS" è in relazione con il fatto che le tavole SI contenute in un TS (attuale) possono informare anche su altri TS.

3.5. Come la Electronic Program Guide utilizza la DVB-SI

In questa sezione spiegheremo come l'utente vede EPG, e come questa utilizza le tavole DVB-SI per arrivare a visualizzare il programma desiderato. Questo ci aiuterà a capire il funzionamento globale e la relazione fra le tavole.

Supponiamo che vogliamo vedere un programma sportivo, per esempio il canale Eurosport.

Scegliamo il Bouquet

Partiamo da una schermata di presentazione della EPG dove si presentano i vari bouquets disponibili (sport, cinema, ...) Vedere Immagine 3.4-1.

Questo menù viene creato a partire dai Bouquet_name_descriptors estratti della BAT.

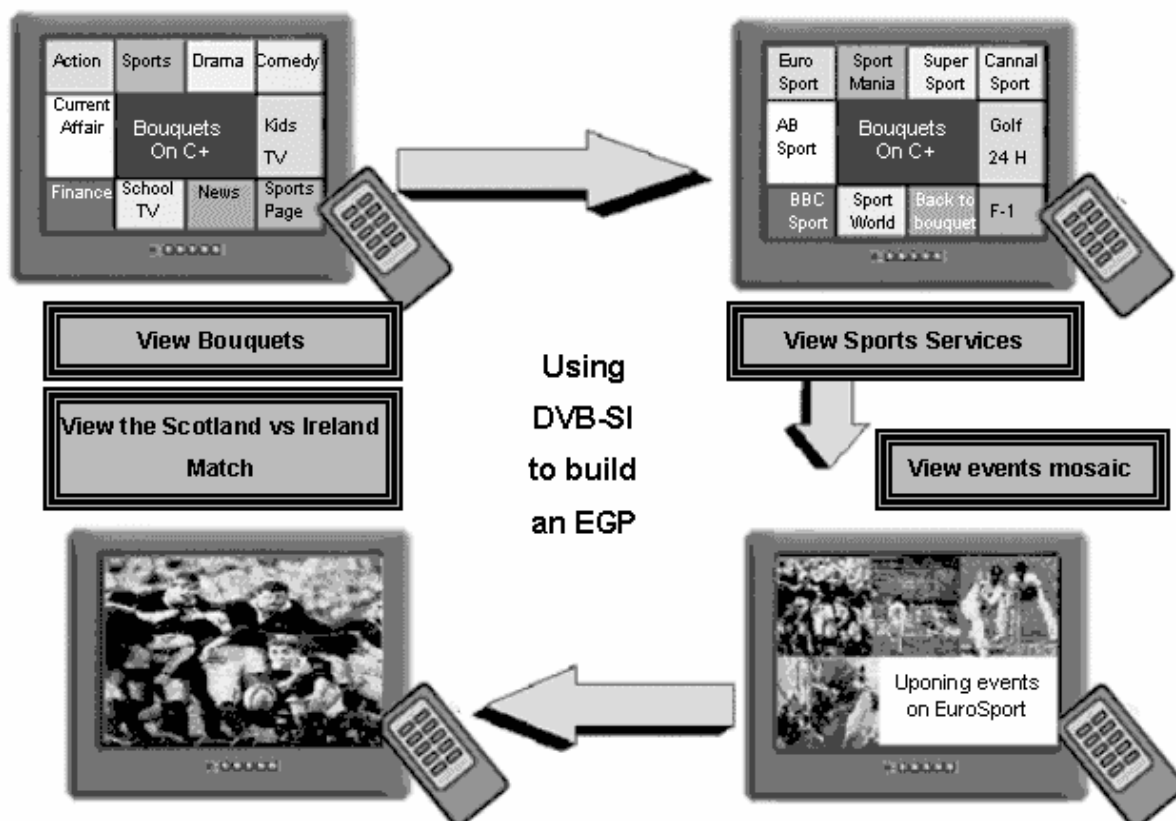
Se qualche bouquet non dovrebbe essere accessibile, per esempio l' Adult Bouquet, questo viene indicato alla BAT attraverso un country_availability_descriptor.

Scegliamo il bouquet dei canali sportivi.

Selezioniamo il servizio Eurosport attraverso la BAT

Una volta scelto un bouquet, a partire dalla BAT associata e dal service_list_descriptor, si visualizza un nuovo menù con i servizi disponibili.

Selezioniamo il servizio Eurosport



Imatge 3.4-1



A questo punto la EPG deve accedere al Transport Stream che trasporta il servizio Eurosport. Per farlo legge il `transport_stream_id` (contenuto nel primo loop della BAT), e quindi accede alla NIT. Nella NIT, cerca i descriptors associati al `transport_stream_id` precedente, e da essi trae le informazioni su frequenza, modulazione, etc .. necessarie affinché l' IRD possa sintonizzare il TS che trasporta il servizio Eurosport.

Vediamo cosa trasmettono su Eurosport, attraverso la SDT

Un volta che l' IRD ha sintonizzato il TS, la EPG, a partire del `service_id` (estratto dal `service_list_descriptor` della BAT) cerca i pacchetti con `PID=11` e `table_id=42`: tavola SDT del TS attuale. Con le informazioni contenute nella SDT ed associata al `service_id` precedente, la EPG ci comunica che il servizio è attivo, e se è presente la EIT Schedule, ci fa informa anche che la programmazione del canale è disponibile.

A partire dal `Mosaic_descriptor` la EPG ci può mostrare una schermata di mosaico con una selezione degli eventi prossimi ed attuali di Eurosport.

Scegliamo un evento Eurosport a partire dalla EIT

Selezioniamo la partita di rugby fra Scozia e Irlanda.

La EPG estrae l'`event_id` dal mosaic e cerca i pacchetti con `PID=12` e con sezioni di `table_id=4E`: EIT Present/Following del TS attuale. Quando ha trovato la sezione EIT che corrisponde al `service_id` precedente, legge i descriptors associati all'`event_id` e visualizza l'orario di inizio, la durata, e informazione testuale relativa all'incontro (squadre, giocatori, ...).

Visione dell'evento e se necessario pagamento per la visione

Una volta accettato di vedere questo evento, a partire dal `service_id` (è identico al `program_number` utilizzato nella PAT), si trova la PMT corrispondente e si avvia il processo di decodificazione nel modo che abbiamo spiegato nella sezione Program Service Information.

Infine attraverso l'`original_network_id` (estrae dalla BAT) e il `telephone_descriptor` (ottenuto dalla SDT), la EPG può effettuare il pagamento dell'evento al provider originale.

Sigle ricorrenti nel testo:

| | |
|-------------|--|
| AAU | Audio Access Unit |
| APU | Audio Presentation Unit |
| AU | Access Unit |
| BAT | Bouquet Association Table |
| CA | Conditional Access |
| CAT | Conditional Access Table |
| CRC | Cyclic Redundancy Check |
| DCT | Discret Cosine Transform |
| DTS | Decoding Time Stamp |
| DVB | Digital Video Broadcasting |
| EBU | European Broadcasting Union |
| EIT | Event Information Table |
| EMM | Entitlement Management Message |
| EPG | Electronic Programme Guide |
| ES | Elementary Stream (ESs = plurale) |
| FEC | Forward Error Correction |
| GOP | Group of Pictures |
| IRD | Integrated Receiver Decoder |
| ISO | International Organization for Standardization |
| MJD | Modified Julian Date |
| MPEG | Moving Pictures Expert Group |
| NIT | Network Information Table |
| NVOD | Near Video On Demand |
| PAT | Program Association Table |
| | |

| | |
|-------------|--|
| PCR | Program Clock Reference |
| PES | Packetized Elementary Stream |
| PESp | Packetized Elementary Stream packets (PESps = plurale) |
| PID | Packet IDentifier |
| PMT | Program Map Table |
| PSI | Program Specific Information |
| PTS | Presentation Time Stamp |
| RS | Reed Solomon |
| RST | Running Status Table |
| SCR | System Clock Reference |
| SDT | Service Description Table |
| SI | Service Information |
| ST | Stuffing Table |
| TDT | Time and Date Table |
| TOT | Time Offset Table |
| TS | Transport Stream |
| TS | Transport Stream |
| TSp | Transport Stream packets (TSps = plurale) |
| UTC | Universal Time, Co-ordinated |
| VAU | Video acces unit |
| VPU | Video Presetation Unit |
| | |
| | |
| | |
| | |