



ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

**CARRERA DE INGENIERÍA EN ELECTRÓNICA Y
TELECOMUNICACIONES**

**TESIS PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERÍA EN
ELECTRÓNICA Y TELECOMUNICACIONES**

AUTOR: MANCHENO ARCOS, CHRISTIAN RICARDO

**TEMA: DESARROLLO DE UNA PLATAFORMA DE GENERACIÓN Y
ANÁLISIS DE FLUJOS DE TRANSPORTE PARA *BROADCAST* (BTS)**

DIRECTOR: DR. GONZALO OLMEDO

CODIRECTOR: ING. FREDDY ACOSTA

SANGOLQUÍ, AGOSTO DEL 2015

Certificado de tutoría

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICADO

Ing. Gonzalo Olmedo PhD.

Ing. Freddy Acosta MSc.

CERTIFICAN

Que el trabajo titulado "DESARROLLO DE UNA PLATAFORMA DE GENERACIÓN Y ANÁLISIS DE FLUJOS DE TRANSPORTE PARA *BROADCAST* (BTS).", realizado por Christian Ricardo Mancheno Arcos, ha sido guiado y revisado periódicamente y cumple normas estatutarias establecidas por la Universidad de las Fuerzas Armadas - ESPE en su reglamento.

Debido a que se trata de un trabajo de investigación recomiendan su publicación.

Sangolquí, Agosto del 2015.

Ing. Gonzalo Olmedo PhD.

DIRECTOR

Ing. Freddy Acosta MSc.

CODIRECTOR

Declaración de Responsabilidad

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

DECLARACIÓN DE RESPONSABILIDAD

CHRISTIAN RICARDO MANCHENO ARCOS

DECLARO QUE:

El proyecto de grado denominado "DESARROLLO DE UNA PLATAFORMA DE GENERACIÓN Y ANÁLISIS DE FLUJOS DE TRANSPORTE PARA *BROADCAST* (BTS)", ha sido desarrollado en base a una investigación exhaustiva, respetando derechos intelectuales de terceros, conforme las citas que constan al pie, de las páginas correspondientes, cuyas fuentes se incorporan en la bibliografía.

Consecuentemente este trabajo es de mi autoría.

En virtud de esta declaración, me responsabilizo del contenido, veracidad y alcance científico del proyecto de grado en mención.

Sangolquí, Agosto del 2015.

Christian Ricardo Mancheno Arcos.

Autorización de publicación

UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE

INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

CHRISTIAN RICARDO MANCHENO ARCOS

Autorizo a la Universidad de las Fuerzas Armadas - ESPE la publicación, en la biblioteca virtual de la Institución del trabajo "DESARROLLO DE UNA PLATAFORMA DE GENERACIÓN Y ANÁLISIS DE FLUJOS DE TRANSPORTE PARA *BROADCAST* (BTS)", cuyo contenido, ideas y criterios son de mi exclusiva responsabilidad y autoría,

Sangolquí, Agosto del 2015.

Christian Ricardo Mancheno Arcos.

DEDICATORIA

El presente proyecto se lo dedico especialmente a mi hijo, Esteban Joaquín, por ser esa personita que llena mis días de felicidad, amor y ternura, quien se ha convertido en la razón para seguir adelante luchando y siendo una mejor persona cada día. También dedico este trabajo a mi madre por ser la principal gestora de este triunfo en vida personal y profesional, me satisface poderte dar esta alegría mami. Por último quiero dedicar este trabajo a Claudia, mi amiga, esposa y compañera por todo lo que ha hecho por mi durante este tiempo que hemos compartimos juntos.

Christian Ricardo Mancheno Arcos.

AGRADECIMIENTO

A mi madre Antonia, que con su esfuerzo, trabajo y dedicación ha hecho de mí una persona de bien, que a pesar de todos los inconvenientes que se presentaron durante este largo y duro camino siempre estuvo ahí para brindarme su apoyo y amor incondicional, por haber confiado en mí y siempre haber sido mi ejemplo a seguir.

Un agradecimiento muy especial a Claudia mi esposa, por su apoyo incondicional, por su dedicación, tiempo y paciencia que ha tenido conmigo durante todo este tiempo.

Agradezco también a toda mi familia, en especial a mis queridas tías quienes siempre han estado ahí, brindándome su amor, comprensión y apoyo, gracias por haberme dado la mano en momentos difíciles, espero alguna vez poder recompensar todo lo que han hecho por mí.

Mis sinceros agradecimientos a todos mis profesores, por todos los conocimientos impartidos en las aulas, al Dr. Gonzalo Olmedo y al Ing. Freddy Acosta por el asesoramiento en el desarrollo de este proyecto, por haberme brindado la oportunidad de trabajar en este proyecto.

Finalmente quiero agradecer a todos mis amigos con quienes compartimos experiencias buenas y malas pero a pesar de todo siempre estuvimos ahí apoyándonos entre todos.

Christian Ricardo Mancheno Arcos

ÍNDICE GENERAL

DEDICATORIA	v
AGRADECIMIENTO	vi
ÍNDICE GENERAL	vii
ÍNDICE DE TABLAS	xi
ÍNDICE DE FIGURAS	xiii
GLOSARIO DE TÉRMINOS	xvi
RESUMEN	xix
ABSTRACT	xx
1 INTRODUCCIÓN	1
1.1 Antecedentes	1
1.2 Justificación e Importancia	2
1.3 Alcance del Proyecto	4
1.4 Objetivos	5
1.4.1 General	5
1.4.2 Específicos	5

2	MARCO TEÓRICO	7
2.1	Televisión digital terrestre	7
2.1.1	Ventajas	7
2.1.2	Estándares	8
2.1.2.1	DVB-T (<i>Digital Video Broadcasting Terrestrial</i>)	8
2.1.2.2	ATSC (<i>Advance Television System Committee</i>)	8
2.1.2.3	DTMB (<i>Digital Terrestrial Multimedia Broadcast</i>)	9
2.1.2.4	ISDB (<i>Integrated Service Digital Broadcasting</i>)	9
2.2	La televisión digital terrestre en el Ecuador	10
2.3	Estándar ISDB-Tb	11
2.3.1	Estructura General	11
2.3.2	Capa física	12
2.3.2.1	Dispersor de energía	13
2.3.2.2	Ajuste de retraso	13
2.3.2.3	Entrelazador	13
2.3.2.4	Codificación Interna	14
2.3.2.5	Codificación externa	14
2.3.2.6	Modulador	15
2.3.2.7	Multiplexación OFDM	15
2.3.3	Flujo de transporte TS	16
2.3.3.1	Encabezado del paquete TS	17
2.3.3.2	Campo de adaptación	18
2.3.3.3	Identificador de paquete (PID)	19
2.3.3.4	Secciones	19
2.3.3.4.1	Estructura de las secciones	21
2.3.3.5	Tablas	22
2.3.3.5.1	Tabla PAT	22

2.3.3.5.2	Tabla PMT	23
2.3.3.5.3	Tabla NIT	26
2.3.4	Remultiplexor	28
2.3.5	Flujo de transporte para <i>broadcast</i> (BTS)	29
2.3.5.1	Asignación del número de paquetes TSP	30
2.3.5.2	Sincronización del BTS	31
2.3.6	Estructura de paquetes TSP, flujo BTS y cuadro múltiplex	34
2.3.6.1	Campo ISDB-Info	35
2.3.6.2	Paquete IIP	39
3	DESARROLLO DEL SOFTWARE	49
3.1	Introducción	49
3.2	Arquitectura de software	50
3.2.1	Arquitectura del Analizador	50
3.2.2	Arquitectura del Generador	51
3.3	Java	53
3.3.1	Archivos en Java	53
3.3.1.1	Clase File	53
3.3.1.2	Clase <i>RandomAccessFile</i>	54
3.4	<i>Analyzer & Generator BTS</i>	56
3.4.1	Estructura del <i>Analyzer & Generator BTS</i>	56
3.4.1.1	Paquete Controlador	57
3.4.1.1.1	Clase AbrirBroadcastTransportStream	57
3.4.1.1.2	Clase AbrirTransportStream	58
3.4.1.1.3	Clase Acciones	58
3.4.1.1.4	Clase GuardarBroadcastTransportStream	58
3.4.1.1.5	Clase LlenarProgressBar	59
3.4.1.1.6	Clase VerificarParametrosISDB	59

3.4.1.2	Paquete Modelo	59
3.4.1.2.1	Clase <i>BroadcastTransportStream</i>	60
3.4.1.2.2	Clase MCCI	60
3.4.1.2.3	Clase TMCC	60
3.4.1.2.4	Clase ISDBInfo	61
3.4.1.2.5	Clase <i>TransportStream</i>	61
3.4.1.3	Paquete Vista	61
3.4.1.3.1	Clase <i>AnalizerGeneratorBTS</i>	61
3.4.1.3.2	Clase IIP	62
3.4.1.3.3	Clase Principal	62
3.4.2	Diagramas de flujo	63
4	PRUEBAS Y RESULTADOS	66
4.1	Pruebas de generación	66
4.2	Pruebas de análisis	71
4.3	Pruebas de transmisión	73
5	Conclusiones y Recomendaciones	75
5.1	Conclusiones	75
5.2	Recomendaciones	76
5.3	Trabajos Futuros	77
	BIBLIOGRAFÍA	78

ÍNDICE DE TABLAS

2.1	Proceso de <i>puncturing</i>	14
2.2	Significado del campo de control de adaptación.	18
2.3	Asignación de los PID.	20
2.4	Tablas PSI/SI.	22
2.5	Tipos de flujos elementales.	25
2.6	Valores de N por segmento.	31
2.7	Cantidad N_{BTS} de paquetes TSP en un Cuadro Múltiplex.	33
2.8	Estructura de la información del campo ISDB-Info.	36
2.9	Estructura del identificador de TMCC.	37
2.10	Estructura del identificador de capa jerárquica.	37
2.11	Indicador de cambio de parámetros de transmisión.	38
2.12	Cantidad de datos AC.	39
2.13	Estructura de la información del paquete IIP.	40
2.14	Palabra de sincronismo.	41
2.15	Multiplexación para datos auxiliares.	41
2.16	Modos del flujo BTS.	42
2.17	Intervalos de guarda del flujo BTS.	42
2.18	Información del TMCC.	43
2.19	Campo de identificación del sistema.	43
2.20	Esquema de modulación del TMCC.	44

2.21	Relación de codificación interna del TMCC.	45
2.22	Entrelazado de tiempo del TMCC.	45
2.23	Número de segmentos en capa capa jerárquica del TMCC.	46
3.1	Constructores de la clase File.	54
3.2	Métodos de la clase File.	54
3.3	Constructores de la clase RandomAccessFile	55
3.4	Métodos de la clase <i>RandomAccessFile</i>	55
4.1	Parámetros de transmisión para el archivo vice.ts	67
4.2	Parámetros de transmisión para el archivo espetv.ts	70
4.3	Parámetros de transmisión para el archivo papaya.ts	71

ÍNDICE DE FIGURAS

2.1 Estructura del sistema de transmisión del Estándar ISDB-Tb.	11
2.2 Diagrama de bloques de la transmisión en ISDB-Tb.	13
2.3 Circuito Dispensor	13
2.4 Diagrama de flujo del codificador convolucional de tasa $1/2$ y $k = 7$	14
2.5 Diagrama de bloques del proceso de modulación	15
2.6 Segmentación del canal de 6 MHz	16
2.7 Estructura de la cabecera del paquete TS	17
2.8 Estructura del campo de adaptación del paquete TS	18
2.9 Estructura de una sección extendida	21
2.10 Tabla PAT	23
2.11 Tabla PMT	24
2.12 Descriptor de información de emergencia	26
2.13 Tabla NIT	27
2.14 Descriptor de recepción parcial	28
2.15 Estructura de los paquetes TSP, flujo BTS y cuadro múltiplex	34
3.1 Diagrama de bloques del Analizador	50
3.2 Diagrama de bloques del Generador	51
3.3 Uso del patrón MVC en el desarrollo de la aplicación.	57
3.4 Diagrama UML clase AbrirBroadcastTransportStream.	57
3.5 Diagrama UML clase AbrirTransportStream.	58

3.6	Diagrama UML clase GuardarBroadcastTransportStream	59
3.7	Diagrama UML clase LlenarProgressBar	59
3.8	Diagrama UML clase VerificarParametrosISDB	59
3.9	Diagrama UML clase BroadcastTransportStream	60
3.10	Diagrama UML clase MCCI	60
3.11	Diagrama UML clase TMCC	61
3.12	Diagrama UML clase ISDBInfo	61
3.13	Diagrama UML clase TransportStream	61
3.14	Interfaz gráfica de usuario del Analizador	62
3.15	Interfaz gráfica de usuario del Generador	62
3.16	Interfaz gráfica del paquete IIP	62
3.17	Diagrama UML clase Principal	62
3.18	Diagrama de flujo del análisis de la PAT y PMT	63
3.19	Diagrama de flujo para la obtención de todos los PID	63
3.20	Diagrama de flujo del proceso de generación	64
3.21	Diagramas de flujo del proceso de búsqueda de paquetes TSP	64
3.22	Diagramas de flujo del proceso de búsqueda de paquetes TSP por PID	64
3.23	Diagramas de flujo del proceso de búsqueda de la información IIP	65
4.1	Selección del <i>Transport Stream</i>	68
4.2	Error en la selección del archivo	68
4.3	Error en los parámetros de configuración.	68
4.4	Configuración de parámetros del <i>Broadcast Transport Stream</i>	69
4.5	Progreso del almacenamiento del <i>Broadcast Transport Stream</i>	69
4.6	Generación del <i>Broadcast Transport Stream</i>	69
4.7	Generación del BTS2.	69
4.8	Configuración de parámetros del BTS2.	70
4.9	Asignación de paquetes para BTS2.	72

4.10	Asignación de los paquetes a la capa A.	72
4.11	Asignación de los paquetes a la capa B.	72
4.12	Asignación de los paquetes a la capa C.	72
4.13	Paquete IIP.	72
4.14	Escenario de pruebas para la transmisión y recepción	73
4.15	Interfaz gráfica de <i>Stream Xpress</i>	73
4.16	Recepción de la señal del <i>Broadcast Transport Stream</i>	74
4.17	Verificación de ancho de banda	74

GLOSARIO

A

AC (*Auxiliary Channel*). Canal auxiliar.

ARIB (*Association of Radio Industrie and Businesses*). Asociación de Radio Industrias y Negocios.

ATSC (*Advanced Television Systems Committee*). Comité de Sistemas de Televisión Avanzados.

B

BTS (*Broadcast Transport Stream*). Flujo de transporte para difusión.

C

CRC (*Cyclic Redundancy Check*). Código de redundancia cíclica.

D

DQPSK (*Diferencial Quadrature Phase Shift Keying*). Modulación de Fase en Cuadratura Diferencial.

DTMB (*Digital Terrestrial Multimedia Broadcasting*). Radiodifusión Digital Terrestre de Multimedia.

DVB-T (*Digital Video Broadcasting Terrestrial*). Radiodifusión de Televisión Digital Terrestre.

DVB-T2 (*Digital Video Broadcasting Terrestrial 2nd Generation*). Radiodifusión de Televisión Digital Terrestre 2da Generación.

G

GINGA *Middleware* Abierto del Sistema Brasileño de Televisión Digital.

GUI (*Graphical User Interface*). Interfaz Gráfica de Usuario

H

HD (*High Definition*). Definición Alta.

I

IDE (*Integrated Development Environment*). Entornos de Desarrollo Integrado

IFFT (*Inverse Fast Fourier Transform*). Transformada inversa rápida de Fourier.

IIP (*ISDB Information Packet*). Paquetes de información ISDB.

ISDB-Tb (*Integrated Services Digital Broadcasting-Terrestrial Brazil*). Servicios Integrados de Radiodifusión Digital - Terrestre Brasileño.

M

MPEG (*Moving Pictures Expert Group*). Grupo de Expertos de Imágenes en Movimiento.

MCCI (*Modulation Control Configuration Information*). Información de modulación, control y configuración.

MVC Modelo Vista Controlador.

N

NIT (*Network Information Table*). Tabla de información de red.

NSI (*Network Synchronization Information*). Información de sincronización de red.

O

OFDM (*Orthogonal Frequency Division Multiplexing*). Multiplexación por División Ortogonal de Frecuencias.

P

PAT (*Program Association Table*). Tabla de asociación de programas.

PCR (*Program Clock Reference*). Referencia de reloj del programa.

PES (*Packet Elementary Stream*). Flujo Elemental Paquetizado.

PID (*Packet Identifier*). Identificador de paquete.

PMT (*Program Map Table*). Tabla de mapeamiento de programas.

PSI (*Program Specific Information*). Información específica de programa.

Q

QAM (*Quadrature Amplitude Modulation*). Modulación de Amplitud en Cuadratura.

QPSK (*Quadrature Phase Shift Keying*). Modulación de Fase en Cuadratura.

R

RS (*Reed Solomon*).

S

SD (*Standar Definition*). Definición Estándar.

SDTV Sistema Brasileño para Televisión Digital.

SI (*Service Information*). Información de servicio.

T

TS (*Transport Stream*). Flujo de transporte.

TMCC (*Transmission and Multiplexing Configuration Control*). Configuración de Control de Transmisión y Multiplexación.

TSP (*Transport Stream Packet*). Paquete de flujo de transporte.

RESUMEN

El presente proyecto comprende el desarrollo de una aplicación para el análisis y generación de los flujos de transporte para difusión (BTS) de la televisión digital terrestre bajo el estándar ISDB-Tb. El *software* desarrollado consta de dos partes, la primera correspondiente al análisis del flujo BTS, permite analizar individualmente cada uno de los paquetes TSP que conforman parte del flujo, mostrando la información correspondiente al campo "ISDB-Info" de cada uno de estos, además permite analizar el contenido del paquete TSP denominado IIP, en especial el contenido de su descriptor conocido como MCCI. La segunda parte correspondiente a la generación del flujo BTS, crea un archivo que contiene el flujo de transporte para difusión conformado de paquetes de 204 bytes de longitud llamados TSP, aquí se da la opción de seleccionar los parámetros de modulación y las configuraciones de las capas jerárquicas, también se puede indicar la capa jerárquica por la cual serán transmitidos los TSP. Se recopiló y estudió la información de la Norma ISDB-Tb necesaria para la elaboración de dicho proyecto, con cual se realizó los algoritmos de programación utilizados en el *software* denominado *Analizer & Generator BTS*, el presente documento contiene toda la información utilizada para la creación de dicho *software*, así como también los diagramas de flujo de los algoritmos utilizados en la programación de la aplicación. Finalmente se realizó pruebas de transmisión de los archivos generados, utilizando televisores que soportan el estándar ISDB-Tb; dichas pruebas fueron realizadas utilizando diferentes archivos generados, los cuales se diferenciaban entre sí por los parámetros de transmisión de las capas jerárquicas que venían configurados en la IIP.

Palabras clave: FLUJO BTS, PAQUETE TSP, ISDB-Info, TMCC, IIP, MCCI.

ABSTRACT

This project was about the development of an application for analysis and generation of Broadcast Transport Streams of terrestrial digital television under the ISDB-Tb standard. The developed software consists of two parts, the first part correspond to the analysis of Broadcast Transport Stream, this allows individually analyze each TSP packages that make up part of the flow, showing the information corresponding to field "ISDB-Info" of each one of these, in addition to analyze the contents of the TSP package called IIP, especially the content of their MCCI descriptor. The second part correspond to the generation of Broadcast Transport Stream, it create a file that contains the Broadcast Transport Stream formed of packets of 204 bytes of length called TSP, here given the option of selecting the modulation parameters and configurations of hierarchical layers, you can also indicate the hierarchical layer which will be transmitted by the TSP. for developing was collected and studied ISDB-Tb information necessary for the development of this project, which scheduling algorithms used in the software called Analyzer & Generator BTS, this document contains all the information used to create the software, as well as the flowcharts of the algorithms used in programming the application. Finally transmission tests performed files was generated using televisions that support the ISDB-Tb standard; these tests were performed using different files generated, which differed from each other by the transmission parameters of hierarchical layers that came configured in the IIP.

Key words: Broadcast Transport Stream, TSP package, ISDB-Info, TMCC, IIP, MCCI.

CAPÍTULO 1

INTRODUCCIÓN

1.1 Antecedentes

A mediados de la segunda década del siglo XXI, la televisión continúa ocupando un lugar de vital importancia en el conjunto de medios de comunicación, debido a su alta penetración y a las grandes sumas de dinero que mueve dicha industria, además del elevado consumo de los contenidos presentados.

La televisión digital, es el producto de la digitalización de los servicios y tecnologías de las distintas versiones de televisión por cable, por satélite o por aire, y representa el cambio tecnológico más radical en la industria televisiva, ya que al codificar sus señales de forma binaria, permite mejorar notablemente la calidad de imagen, el efecto de sonido y sobre todo el uso optimizado del espectro radioeléctrico, además habilita la posibilidad de crear vías de retorno entre consumidor y productor de contenidos, lo que abre la opción de crear aplicaciones interactivas, haciendo del televisor un terminal multimedia de mejores características [1].

Actualmente existen cuatro normas de televisión digital terrestre definidas por Estados Unidos, Europa, Japón y China, las cuales han sido o están siendo adoptadas en varios países del mundo dependiendo de las posibilidades tecnológicas que ofrecen cada una de ellas, además obedeciendo a intereses políticos y económicos de cada nación. Específicamente en Ecuador, el 25 de marzo de 2010, el CONATEL re-

solvió adoptar el estándar japonés-brasileño ISDB-Tb a través de su resolución 084-05 CONATEL-2010 [2].

El grupo de investigación de televisión digital de La Universidad de las Fuerzas Armadas - ESPE , actualmente cuenta con un laboratorio donde se realizan pruebas de generación de contenidos, transmisión y recepción de señales bajo el estándar ISDB-Tb. Para la configuración automática del modulador ISDB-Tb es necesario extraer los parámetros de configuración incorporad en el flujo de transporte para difusión conocido como BTS, el cual adicionalmente contiene los paquetes multiplexados de video, audio y datos y la asignación de las capas jerárquicas para cada uno de ellos.

La Universidad de las Fuerzas Armadas - ESPE adquirió la plataforma *Village-Flow*, que tiene entre una de sus funciones principales la remultiplexación de paquetes TS para la obtención del *Broadcast Transport Stream* (BTS). Este *software* funciona con el sistema operativo *Windows* y presenta como funcionalidad principal el hecho que permite la transmisión en vivo de uno o varios programas de televisión, puesto que realiza la codificación y multiplexación de la señales internamente [3].

Dentro de los sistemas de software libre existen únicamente analizadores de *Transport Stream* y los de *Broadcast Transport Stream* pasan a ser parte de equipos o programas específicos con costos muy elevados. Adicionalmente es importante mencionar que al crear la señal BTS se podrá crear transmisores de propiedad intelectual de ES-PETV en tiempo real, con mejores prestaciones que otras plataformas.

1.2 Justificación e Importancia

Los altos costos económicos que representa implementar un sistema de transmisión de televisión digital, han obligado a que se planteen soluciones de *software* alternativas, que brinden las mismas prestaciones que las ofertadas en el mercado y que permi-

tan disminuir los costos de los sistemas de transmisión, consiguiendo así disminuir los equipos de *hardware* utilizados para realizar la codificación, modulación y multiplexación de la señal de televisión. Las herramientas más importantes que conforman un sistema de transmisión basado en *software* son una tarjeta moduladora, responsable de la etapa de transmisión, y un software como *VillageFlow*, encargado de la etapa de codificación y multiplexación del sistema de televisión digital [1].

La Universidad de las Fuerzas Armadas - ESPE cuenta con la tarjeta moduladora DekTec DTU-215 y además con las licencias y el kit de desarrollo de *software*, del programa propietario necesario para el control de dicha tarjeta conocido como *DTC-300-SP StreamXpress*, el cual permite leer tramas de *Transport Stream* (TS) y *Broadcast Transport Stream* (BTS), configurar las características de capa física para el caso de un *Transport Stream* y transmitir a través del puerto de salida de la tarjeta moduladora *DekTec*. La ESPE también dispone de las licencias del *software* propietario *VillageFlow*, el cual específicamente realiza funciones de codificación, multiplexación, remultiplexación y modulación; todo esto en conjunto constituye el esquema básico de solución de transmisión, basado en software con el que cuenta la ESPE [3].

El desarrollo de este proyecto, se centra justamente en realizar un *software* de propiedad intelectual de ESPETV, el cual reemplace las funciones de uno de los componentes de dicho sistema de transmisión, es decir que permita multiplexar el contenido de audio, video y datos encapsulado en las tramas del flujo de transporte TS, para generar las tramas que conforman el flujo de transporte para difusión. He aquí la importancia del proyecto ya que mediante el desarrollo de dicha aplicación, la cual formará parte del sistema de transmisión de televisión digital que se está desarrollando por parte del grupo de investigación ESPETV, se podrá crear un transmisor de propiedad intelectual de ESPETV, el cual permita modificar los 16 bytes de control de transmisión que contienen la configuración de los segmentos del canal y paráme-

tros de transmisión [4]; además dicha aplicación también permitirá realizar el análisis de flujos BTS mostrando la información de los 16 bytes de control de transmisión, lo cual es una característica muy importante de la aplicación ya que en la actualidad no existe ningún *software* gratuito que permita visualizar la información de los 16 bytes adicionales de los paquetes TSP.

1.3 Alcance del Proyecto

El presente proyecto conforma una de las etapas del sistema de transmisión de Televisión Digital que se está desarrollando por parte del grupo de investigación ESPETV, el cual permitirá a los usuarios del laboratorio de Televisión Digital de la ESPE dejar de depender de *software* licenciados con precios elevados, necesarios para la transmisión de la señal de televisión.

Esta aplicación se centra básicamente en analizar detalladamente los 16 bytes agregados a cada paquete del *Transport Stream*, en la obtención del flujo de transporte de difusión BTS, al momento de la remultiplexación de los paquetes TS que transportan los distintos programas o servicios. Además en su etapa de análisis, dicha aplicación permitirá visualizar la información transportada en el denominado paquete IIP, la cual nos mostrará principalmente la información de configuración de control y modulación de cada una de las capas jerárquicas.

Posteriormente en su etapa de generación la aplicación creará un archivo que contenga el *Broadcast Transport Stream*, permitiendo modificar los parámetros de las 3 capas jerárquicas, el modo, el intervalo de guarda y la programación de los segmentos, para cada BTS que se desee generar.

También se realizarán pruebas de transmisión de los diferentes flujos BTS generados para verificar el correcto funcionamiento del *software* desarrollado; el escenario de

dichas pruebas estará conformado por los equipos con los que actualmente cuenta el Laboratorio de televisión digital y el *software* resultante, ya que actualmente el sistema de transmisión del grupo ESPETV que se mencionó anteriormente, aún está en etapa de desarrollo y al ser este proyecto solo una parte de dicho sistema no se dependerá de la ejecución de las partes restantes para la culminación del mismo.

1.4 Objetivos

1.4.1 General

- Desarrollar un software en un lenguaje de código abierto, que permita analizar y generar los paquetes TSP, que conforman el *Broadcast Transport Stream*, de televisión digital terrestre bajo la norma ISDB-Tb.

1.4.2 Específicos

- Analizar el paquete IIP y los campos "ISDB-Info" y "RS", de cada uno de los paquetes TSP, para establecer toda la información del flujo BTS.
- Generar el paquete IIP y campos "ISDB-Info" y "RS", agregados al momento de la multiplexación de los diferentes TS, para obtener el flujo BTS.
- Obtener la tasa binaria de las tablas y flujo elementales presentes en *Transport Stream*.
- Modificar el modo, intervalo de guarda, los parámetros de las capas jerárquicas y la asignación de paquetes, para obtener diferentes configuraciones de los TSP y del IIP que conforman el flujo BTS.
- Obtener la lista de programas y flujo elementales que conforman el *Transport Stream*, cada uno con su respectivo PID.

- Investigar la función de las tablas que conforman el *Transport Stream* y determinar cuales contienen información útil para la elaboración del *software*.
- Realizar pruebas de transmisión del flujo BTS generado y validar el *software* desarrollado, mediante la correcta recepción de dichos flujos.
- Elaborar una interfaz gráfica de usuario (GUI) amigable y fácil de utilizar.

CAPÍTULO 2

MARCO TEÓRICO

2.1 Televisión digital terrestre

La televisión digital terrestre (TDT), es el conjunto de tecnologías que transiten y reciben las señales de audio y video por medio de señales digitales, utilizando como medio de transmisión frecuencias del espectro radioeléctrico.

Una de las ventajas de la televisión digital con respecto a la analógica, es que la digital codifica las señales a transmitirse en forma de bits permitiendo de esta manera, tener una mejor calidad de video y audio, optimizar el espectro radioeléctrico y crear aplicaciones interactivas en las cuales el consumidor pueda interactuar con el productor de contenidos [5].

2.1.1 Ventajas

Además de la codificación binaria de señales, la TDT presenta algunas ventajas más sobre la televisión analógica:

- **Calidad:** Mejor calidad de audio, mayor nitidez y mejor resolución de la imagen.
- **Interactividad:** Integración de contenidos, a través de servicios públicos y comerciales.

- **Movilidad:** Recepción de la señales de televisión en dispositivos móviles y portátiles.
- **Desarrollo:** Penetración del servicio a zonas rurales con el uso de una única frecuencia.
- **Optimización del uso del espectro:** Transmisión de varios programas en diferentes definiciones, en el mismo canal de 6 MHz.

2.1.2 Estándares

Actualmente en el mundo existen 4 estándares de televisión digital terrestre: DVB-T, ATSC, ISDB-t, DTMB definidos por Europa, Estados Unidos, Japón y China respectivamente, y dos variantes para dos de ellos, una para el estándar Japonés y otra para el estándar Europeo, conocidos como ISDB-Tb y DVB-T2 respectivamente [6].

2.1.2.1 DVB-T (*Digital Video Broadcasting Terrestrial*)

Fue creado por países de la unión europea y permite transmitir varias señales de definición estándar (SD) en un mismo canal. Aunque el sistema fue diseñado para canales de 8 MHz, también puede ser usado en canales de 6 y 7 MHz, teniendo en cuenta que al hacer esto disminuirá la capacidad de transmisión del sistema. Utiliza el flujo de transporte MPEG-2 como señal de entrada normalizada, y OFDM como esquema de modulación para la transmisión.

Ha sido adoptado en varios países de Europa, Asia, África, Oceanía y América, entre los cuales destacan Guayana Francesa, Uruguay, Colombia y Panamá.

2.1.2.2 ATSC (*Advance Television System Committee*)

Fue creado por Estados Unidos para reemplazar al sistema tradicional de televisión analógica NTSC y permite la transmisión de una señal de TDT en alta definición,

sin dejar de lado la transmisión de señales de definición estándar, ya que también permite transmitir varias señales SD. Fue diseñado para canales de 6 MHz, utilizando codificación de video MPEG-2 y modulación 8-VBS, la cual evita de mejor forma las distorsiones.

Ha sido adoptado en Estados Unidos, Canadá, México, Corea del Sur, Honduras y El Salvador.

2.1.2.3 DTMB (*Digital Terrestrial Multimedia Broadcast*)

Fue creado por China como la fusión de otros dos estándares, utiliza TDS-OFDM (*Time Domain Synchronization - Orthogonal Frequency Division Multiplexing*) como esquema de modulación y LDPC (*Low-Density Parity-Check*) como esquema de codificación de canal; además permite la recepción fija y móvil y la combinación de HD, SD y servicios multimedia.

2.1.2.4 ISDB (*Integrated Service Digital Broadcasting*)

Fue creado por *Association of Radio Industries and Businesses* (ARIB) y adoptado en Japón, su principal característica es que divide el canal 6 MHz en trece segmentos, permitiendo seleccionar la combinación de dichos segmentos. Utiliza la codificación de video MPEG-2 y OFDM-BTS (*Orthogonal Frequency Division Multiplex - Band Segmented Transmission*) como esquema de modulación; permite la recepción de la señal de televisión en terminales móviles bajo el concepto de recepción parcial de un segmento conocida "*one-seg*", utiliza un canal de 6 MHz de ancho de banda dividido en 13 segmentos y habilita la transmisión jerárquica permitiendo transmitir la información en tres capas jerárquicas conocidas como A, B y C. En base a este estándar, se crea el estándar brasileño conocido como SBTVDT (*Sistema Brasileiro de Televisión Digital Terrestre*) ó ISDB-Tb, que utiliza diferente compresión de audio y video, *middleware* y protección de contenido que el estándar japonés [7].

Ha sido adoptado en Brasil, Botsuana, Japón, Argentina, Filipinas, Ecuador, Maldivas, Perú, Sri Lanka, Chile, Venezuela, Costa Rica, Paraguay, Bolivia, Nicaragua, Uruguay, Guatemala y Honduras.

2.2 La televisión digital terrestre en el Ecuador

El 25 de marzo de 2010, Ecuador adoptó oficialmente el estándar japonés-brasileño (ISDB-Tb) para la Televisión Digital Terrestre [2]. Desde entonces el Ministerio de telecomunicaciones junto con las entidades públicas y privadas del país han trabajado en la implementación de dicho sistema en el país; teniendo como resultados la emisión al aire de la primera señal digital de televisión correspondiente a TC Televisión en el 2013.

Actualmente en el país existen 519 estaciones de TV abierta entre matrices y repetidoras que requieren digitalizarse, para lo cual se ha dispuesto un calendario del apagón analógico en el país previsto a realizarse en 3 fases. La fase 1 que tiene como fecha límite el 31 de diciembre del 2016, propone realizar el apagón analógico en áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población mayor a 500.000 habitantes que incluiría a Quito, Guayaquil y Cuenca. La fase 2 que tiene como fecha límite el 31 de diciembre del 2017, propone realizar el apagón analógico en áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población entre 500.000 y 200.000 habitantes que incluiría a las capitales de las provincias. La fase 3 que tiene como fecha límite el 31 de diciembre del 2018, propone realizar el apagón analógico en áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población menor a 200.000 habitantes que incluiría al resto del país [8].

Así como el sistema digital de televisión presenta una serie de ventajas frente al

analógico, también hay que considerar que tiene desventajas muy considerables tanto para los televidentes como para las televisoras; entre las principales de ellas destacan la adquisición de nuevos televisores que soporten el estándar o a su vez de decodificadores que conviertan la señal, y la desaparición de televisoras pequeñas que no tengan el suficiente poder económico para afrontar el cambio digital.

Entre los próximos retos a seguir en Ecuador por parte del gobierno están la entrega de decodificadores a la población beneficiaria, la entrega de concesiones definitivas de licencias para TDT, la implementación del sistema de alerta de emergencia, la implementación de interactividad y la generación educativos, culturales y de salud, los cuales se irán realizando en los próximos años a la par del apagón analógico.

2.3 Estándar ISDB-Tb

El estándar conocido también como SBTVD-T fue desarrollado por Brasil en el año 2006, luego de introducir algunas modificaciones al estándar japonés ISDB-Tb, entre las cuales se destacan la adopción de MPEG-4 para la compresión de datos, la introducción de un nuevo *Middleware* conocido como *Ginga* mucho más robusto que el *Middleware BML* usado por el estándar ISDB-T, la tasa de presentación de 30 cuadros/s incluso para receptores móviles y la implementación de libre recepción en lugar de *B-CAS DRM Copy protection* usada en ISDB-T.

2.3.1 Estructura General

En la Figura 2.1 se observa la estructura del sistema de transmisión del estándar ISDB-Tb, compuesta de 3 bloques: Bloque de código fuente, bloque de multiplexado y bloque de transmisión.

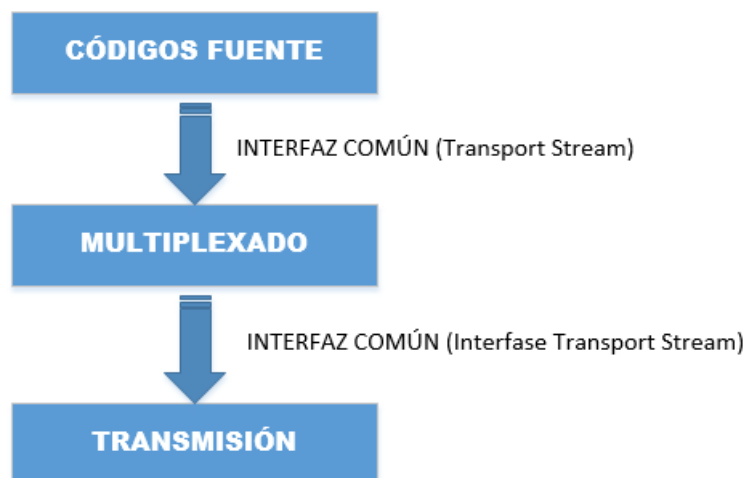


Figura 2.1: Estructura del sistema de transmisión del Estándar ISDB-Tb.

En el bloque de códigos fuente el estándar ISDB-TB utiliza MPEG-4 y H.264 para la codificación de video de recepción fija y móvil respectivamente, mientras que para la codificación de audio utiliza HE-ACC tanto para recepción fija y móvil; para la codificación de datos se utiliza la plataforma *Ginja*, la cual es un *middleware* propio del sistema.

En el bloque de multiplexación, el estándar multiplexa los contenidos de audio, video y datos en un único flujo de transporte conocido como *Transport Stream*, el cual se obtiene a partir de los paquetes PES *Packet Elementary Stream* en un etapa previa.

En el bloque de transmisión el estándar especifica el tipo de modulación y codificación a ser utilizadas.

2.3.2 Capa física

La capa física de ISDB-Tb brinda una alta calidad en la transmisión de video, audio y datos, para lo cual cada uno de los paquetes asignados a cada capas jerárquicas en la etapa de remultiplexación, deben atravesar por una serie de procesos , descritos en la Figura 2.2. Además de lo mencionado anteriormente la capa física del estándar cumple

con las siguientes funciones:

- Soporta transmisiones de tramas codificadas, en ambientes propensos a errores.
- Soporta la multiplexación de varios programas en un único flujo de transporte.
- Soporta condiciones de acceso.
- Garantiza que los datos que ingresan al buffer no se pierdan.
- Maneja información privada de los clientes.
- Añade y extrae programas de acuerdo a la programación local [9].

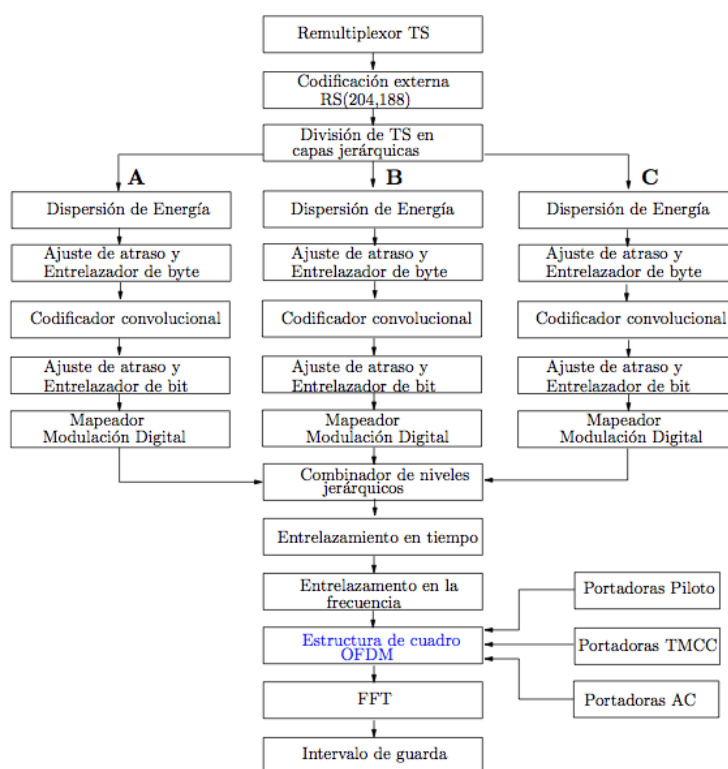


Figura 2.2: Diagrama de bloques de la transmisión en ISDB-Tb.

A continuación se describen cada uno de los bloques por los cuales deben atravesar cada uno de los paquetes de cada capa.

2.3.2.1 Dispersor de energía

Este proceso ocurre en cada una de las capas jerárquicas, y tiene la función de evitar una secuencia de transmisión sucesiva de ceros o unos, para lo cual ejecuta una XOR entre los datos de cada capa y una secuencia aleatoria generada por el polinomio $g(x) = x^{15} + x^{14} + 1$, que inicialmente toma el valor 1 0 0 1 0 1 0 1 0 0 0 0 0 0 0.

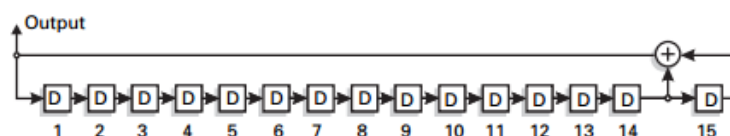


Figura 2.3: Circuito Dispersor

2.3.2.2 Ajuste de retraso

Las diferentes configuraciones realizadas en cada capa jerárquica causa diferentes retrasos en el receptor, para lo cual se ejecuta una corrección de retraso en el proceso de la transmisión. El ajuste se realiza mediante la inserción de diferentes retrasos en las capas de acuerdo con el esquema de modulación y la tasa de código del codificador interno [10].

2.3.2.3 Entrelazador

Es el responsable de la mezcla de los bytes que forman los TSP, para evitar que los errores introducidos por el canal de transmisión no puedan ser corregidos por los códigos correctores de errores.

2.3.2.4 Codificación Interna

La codificación interna es realizada mediante un codificador convolucional, que además se encarga de la protección de datos y la de bits redundantes como se muestra en la Figura 2.4.

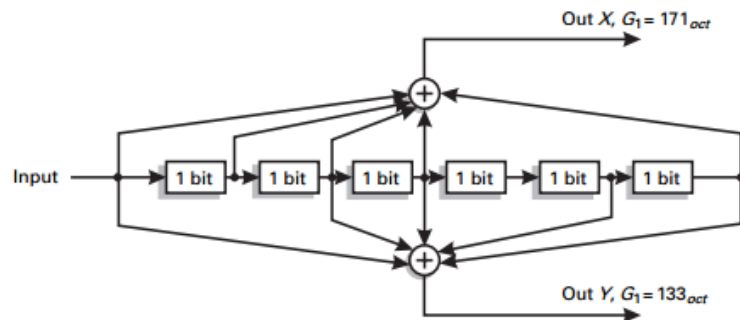


Figura 2.4: Diagrama de flujo del codificador convolucional de tasa $1/2 = y$ $k = 7$

El diagrama del codificador utilizado tiene una longitud de restricción $K = 7$, y genera un código con tasa $1/2$; además contiene 2 salidas, cada una formada por la unión de las diferentes etapas del registro de desplazamiento, que se definen por la matriz generadora $G = [171; 133]$. Este esquema de codificación permite la producción de códigos con diferentes tasas, para lo cual utiliza un esquema de supresión de bits conocido como *puncturing*, descrito en la Tabla 2.1.

Tabla 2.1: Proceso de *puncturing*.

Tasa de código	Puncturing	Secuencia de transmisión
$1/2$	X:1, Y:1	X_1, Y_1
$2/3$	X:10, Y:11	X_1, Y_1, Y_2
$3/4$	X:101, Y:110	X_1, Y_1, Y_2, X_3
$5/6$	X:10101, Y:11010	$X_1, Y_1, Y_2, X_3, Y_4, X_5$
$7/8$	X:10000101, Y:1111010	$X_1, Y_1, Y_2, X_3, Y_4, X_5, Y_6, Y_7$

2.3.2.5 Codificación externa

Luego de la etapa de remultiplexación los campos ISDB-Info y RS de cada TSP son sustituidos por los 16 bytes de paridad del código *Reed Solomon* (RS) utilizado, que permite corregir hasta 8 bytes erróneos.

2.3.2.6 Modulador

El modulador es el encargado de transformar los bits codificados en ondas electromagnéticas para poder ser transmitidas. El modulador ISDB-T incluye varios bloques mostrados en la Figura 2.5, los cuales realizan diferentes operaciones.

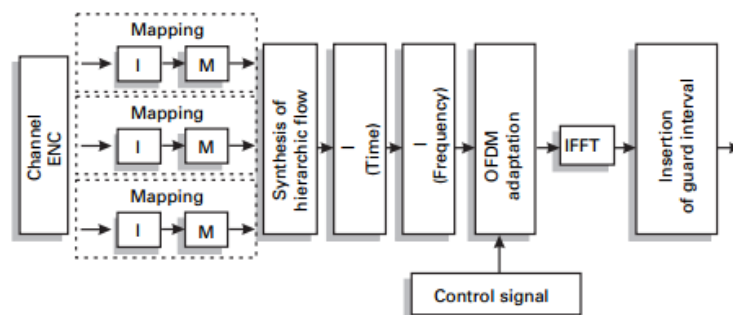


Figura 2.5: Diagrama de bloques del proceso de modulación

Los bits que ingresan al modulador son sometidos a un proceso de entrelazado y luego son mapeados en una constelación de modulación digital. Los esquemas de modulación usados para la transmisión de datos en el estándar ISDB-T son: DQPSK, QPSK, 16-QAM Y 64-QAM.

En la etapa de entrelazado de tiempo se insertan retrasos entre las portadoras de segmentos para evitar la transmisión de una secuencia de datos al mismo tiempo, incluso si están en diferentes portadoras; mientras que en la etapa de entrelazado de frecuencia se ejecuta un proceso conocido como *shuffling* entre las portadoras del mismo segmento, con el fin de conseguir un ordenamiento aleatorio de frecuencias [9].

2.3.2.7 Multiplexación OFDM

La multiplexación *OFDM* es utilizada como esquema de transmisión, con el fin de transmitir señales digitales ahorrando espectro radioeléctrico; esto se consigue gracias a la ortogonalidad que existen entre las subportadoras, permitiendo que el espectro de

la señal OFDM sea lo más estrecho posible y a su vez hace que ajuste la señal en el canal de 6 MHz.

El estándar utiliza BTS-OFDM que es una versión segmentada de OFDM, dicho esquema utiliza un conjunto de bloques de frecuencias básicas denominadas segmentos, que tienen una estructura de soporte común. La asignación de segmentos del estándar ISDB-Tb en el ancho de banda de 6 MHz se muestra en la Figura 2.6.

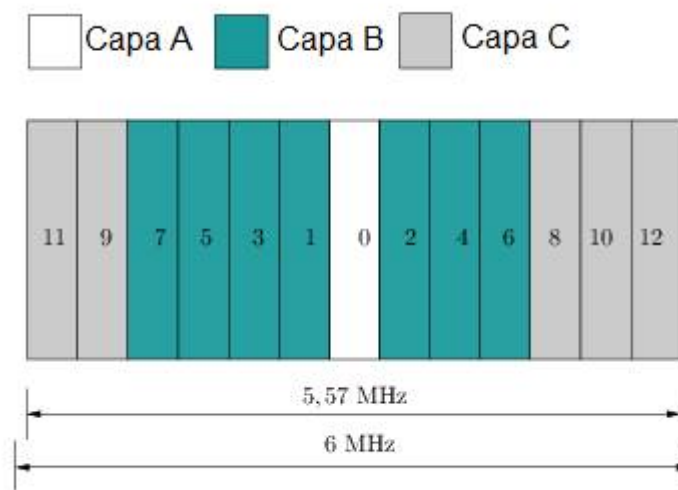


Figura 2.6: Segmentación del canal de 6 MHz

De la Figura 2.6 se puede observar que la segmentación del canal produce un total de 14 segmentos, de los cuales solo 13 son utilizables, ya que uno de ellos es distribuido entre las bandas de guarda superior e inferior del canal [11]. El ancho de banda de cada canal se lo puede obtener utilizando la ecuación 2.1.

$$BW = \frac{6}{14} = 428,57kHz \quad (2.1)$$

En los sistemas *OFDM*, es necesario introducir un tiempo antes de la transmisión de cada símbolo, el cual es conocido como intervalo de guarda; su función es eliminar

o disminuir la interferencia entre los símbolos provocada por las replicas de la señal principal, que tienen un retardo menor que el intervalo de guarda. Los valores de tiempo de guarda son: $1/4T$, $1/8T$, $1/16T$ y $1/32T$ que serán asignados de acuerdo al ambiente de transmisión.

2.3.3 Flujo de transporte TS

El TS (*Transport Stream*) es un flujo en serie que contiene los paquetes de audio, video y datos codificados correspondientes a los programas que se desea emitir. Este flujo está formado de paquetes de 188 bytes conocidos como paquetes TS.

El TS se obtiene a partir de paquetes PES (*Packetized Elementary Stream*) de longitud fija y se adapta a las necesidades de TDT, ya que permite la multiprogramación sin necesidad de contar con una referencia de reloj común entre todos los programas transportados.

Dentro de los 188 bytes de cada paquete TS se incluye una cabecera de 4 bytes que cumple con las funciones de señalar el inicio de cada paquete e identificar el tipo de contenido transportado mediante el campo PID (*Packet Identifier*). Luego de la cabecera se encuentra el campo de carga útil que en ocasiones puede dividirse en dos partes, incorporando el campo de adaptación que se utiliza cuando el encabezado del paquete es mayor a 4 bytes [12].

2.3.3.1 Encabezado del paquete TS

La estructura de la cabecera del paquete TS se muestra en la Figura 2.7. A continuación se describe la función de cada uno de los campos de la Figura 2.7.

- **sync**: preámbulo de sincronización, siempre tiene el valor de 47_H .
- **transport error indicator**: este bit se activa cuando existe un error dentro del

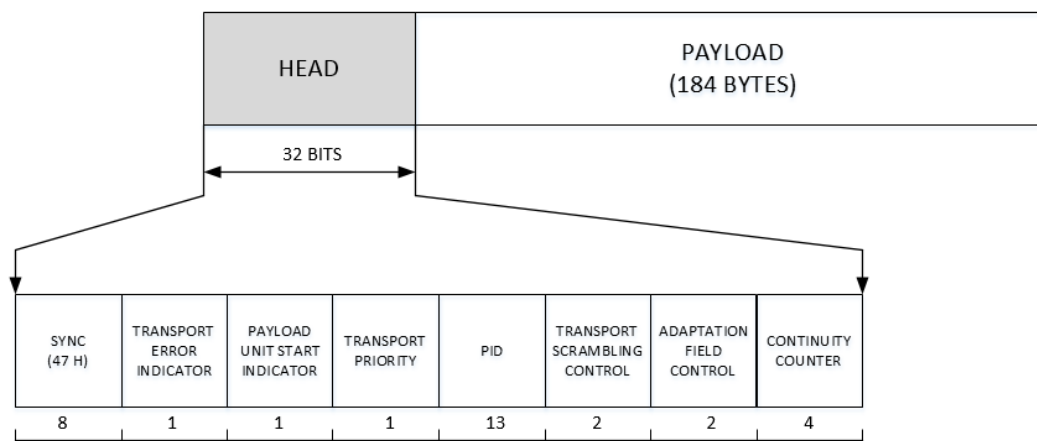


Figura 2.7: Estructura de la cabecera del paquete TS

paquete.

- **payload unit start indicator:** este bit se activa cuando en el paquete se transportan tablas o un PES. En el caso de transportar tablas, el primer byte del campo de carga útil corresponde a un indicador que apunta al comienzo de la primera sección, caso contrario este byte coincide con el inicio del paquete PES.
- **transport priority:** este bit se activa para indicar una mayor prioridad del paquete, con respecto a los paquete con el mismo PID.
- **PID (Packet identifier):** este campo de 13 bits permite identificar los paquetes con diferentes flujos ES.
- **transport scrambling control:** este campo de 2 bits indica el modo de codificación de los bytes de carga útil.
- **adaptation field control:** este campo de 2 bits indica si en dicho paquete se transporta los campos de adaptación y carga útil, de acuerdo a la Tabla 2.2.
- **continuity counter:** este campo de 4 bits es un contador de paquete que tienen el mismo PID, inicia en 0000 y cuando alcanza su máximo valor 1111 se reinicia el ciclo.

Tabla 2.2: Significado del campo de control de adaptación.

Valor	Significado
00	Reservado
01	Solamente carga útil
10	Solamente campo de adaptación
11	Campo de adaptación y carga útil

2.3.3.2 Campo de adaptación

El campo de adaptación esta presente solo en algunos paquetes TS, dependiendo del valor que tome el *adaptation field control*, mostrados en la Tabla 2.2. Cumple con la función de relleno para mantener constantes los 188 bytes del paquete TS en ciertas ocasiones, principalmente cuando un paquete transporta el denominado PCR (*Program Clock Reference*), que es utilizado en el receptor para sincronizar los todos los paquetes del flujo TS [11].

A continuación se describe la función de los campos más relevantes de la Figura 2.8.

- ***adaptation field length:*** este campo de 8 bits indica la longitud en bytes del campo de adaptación, contando a partir del siguiente byte a este.
- ***discontinuity indicator:*** este bit se activa cuando existe alguna discontinuidad.
- ***stuffing bytes:*** son bytes de relleno usados para formar el paquete de 188 bytes.
- ***splice countdown:*** este campo de 8 bits permite una correcta conmutación entre un TS y otro.
- **PCR:** Transporta la referencia del reloj principal, ocupa un campo de 42 bits, de los cuales 33 bits son la base y 9 bits son la extensión.

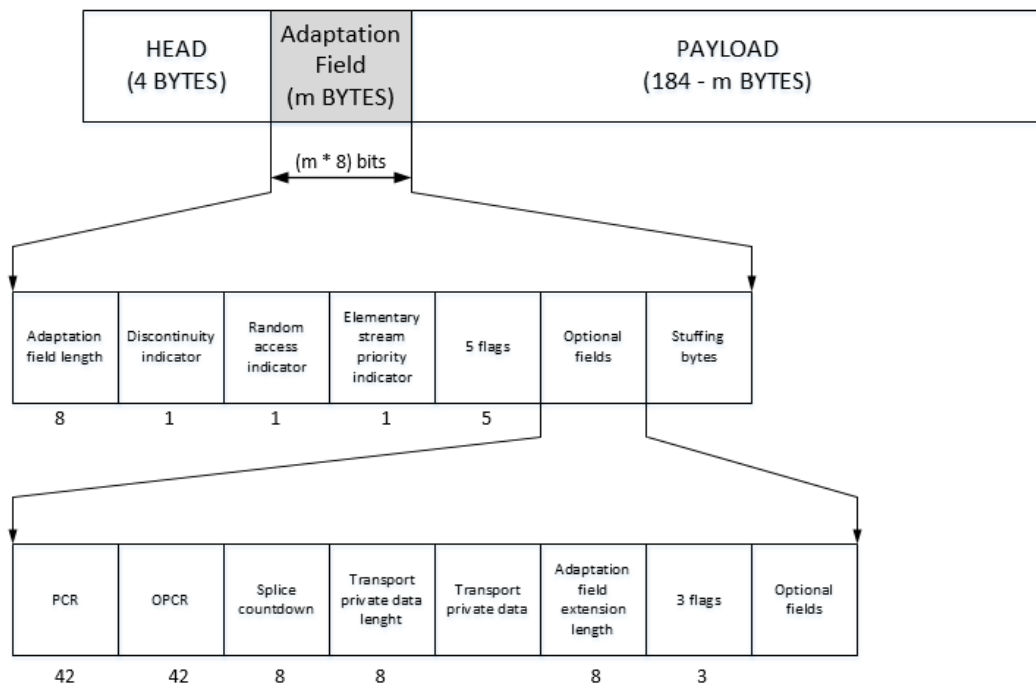


Figura 2.8: Estructura del campo de adaptación del paquete TS

2.3.3.3 Identificador de paquete (PID)

EL campo PID de los paquetes TS se utiliza para identificar el tipo de *stream* que transporta cada paquete. Permite asignar 8192 valores distintos a los flujos ES que forman parte del flujo TS, de los cuales los primeros 17 están reservados para las algunas tablas definidas por la norma de transporte MPEG-2. El valor $1FFF_H$ (8191) está reservado para los paquetes nulos. En la Tabla 2.3 se presentan las asignaciones de los PID más utilizados.

2.3.3.4 Secciones

Una sección es una estructura que debe emplearse para insertar las tablas en un paquete TS. A diferencia de los paquetes PES transportados en un paquete TS, las secciones de las tablas PSI y SI no deben empezar ni acabar obligatoriamente con un paquete TS [12]. De ser el caso que un paquete TS contiene el comienzo de al menos una sección el campo *payload unit start indicator* debe ponerse a 1, caso contrario si ninguna sección

Tabla 2.3: Asignación de los PID.

Valores PID		Asignación
Hex	Decimal	
0000	0	Tabla PAT
0001	1	Tabla CAT
0010	16	Tabla NIT
0011	17	Tabla BAT
0011	17	Tabla SDT
0012	18	Tabla EIT
0026	38	
0027	39	
0014	20	Tabla TDT
0014	20	Tabla TOT
0000	0	Tabla ST
1FFF	8191	Paquetes Nulos

tiene su comienzo dentro del paquete TS el *payload unit start indicator* se pondrá a 0.

La longitud de las secciones es variable y está limitada a 1024 bytes con excepción de las secciones de la tabla EIT que puede alcanzar 4096 bytes [12]. Si una sección termina antes del final de un paquete TS, es posible incluir hasta 10 secciones como máximo en el mismo paquete, aunque no es muy recomendable insertar más de una sección dentro del mismo paquete, para lo cual se utiliza un método de relleno en el cual se colocan a 0xFF todos los bytes sobrantes.

Cuando en un paquete TS se transporta una sección, el primer byte del *payload* es el *pointer field* que indica la cantidad de bytes que existen desde el final de este campo hasta el primer byte de dicha sección, es decir un valor de 00_H indica que la sección

empieza inmediatamente después del *pointer field*. Cabe recalcar que el *pointer field* sirve exclusivamente para señalar la posición de la primera sección que contenga su comienzo dentro del paquete TS, es decir si ninguna sección tiene su comienzo dentro del paquete TS, no existirá dicho campo.

2.3.3.4.1 Estructura de las secciones

Cada sección tiene un encabezado y un campo de carga útil, que empieza con un campo denominado *table id* y termina con un código de redundancia cíclica (CRC32). A continuación en la Figura 2.9, se muestra la estructura de una sección extendida que es generalmente la que se utiliza en la mayoría de las tablas de la norma ISDB-Tb, la diferencia que presenta esta con respecto a una sección general es los campos del encabezado, que en el caso de una sección general va desde el *table id* hasta *section length*.

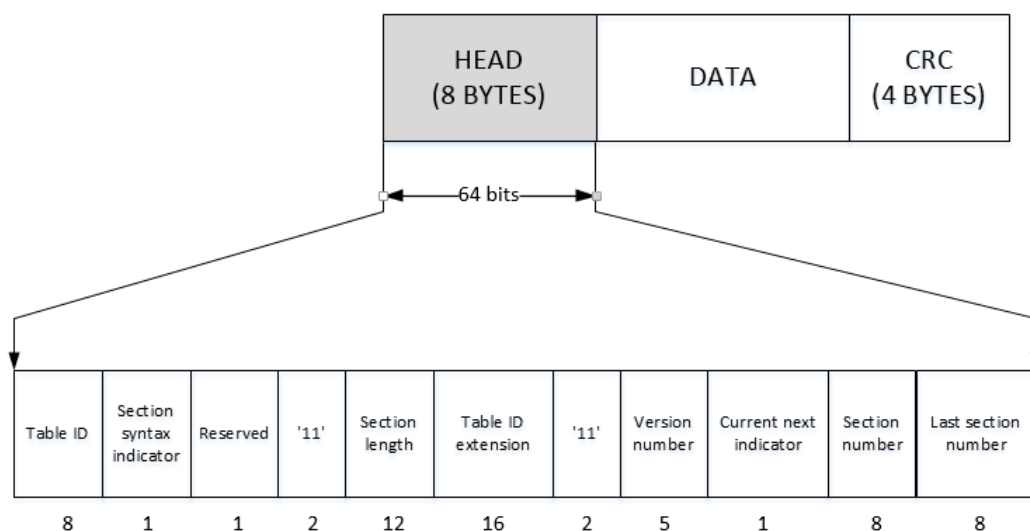


Figura 2.9: Estructura de un sección extendida

La función de los campos del encabezado de la Figura 2.9 es la siguiente.

- **table ID:** es un identificador de cada tabla.

- **section syntax indicator:** este bit se activa cuando se trata de una sección extendida, caso contrario permanece en cero.
- **reserved:** este bit se activa cuando se trata de una sección privada, caso contrario permanece en cero.
- **section length:** este campo de 12 bits indica el tamaño en bytes de la sección, contando a partir del siguiente byte a este, hasta el final de la sección. Los dos primeros bits siempre tienen el valor 00.
- **table ID extension:** es una extensión opcional del table ID.
- **version number:** este campo de 5 bits indica el número de la sección, cada que se altera un parámetro de algún parámetro de una tabla, este campo se altera.
- **current next indicator:** este bit permanece activo siempre y cuando la tabla sea válida, de lo contrario se pone a 0 para indicar que se debe descartar la tabla.
- **section number:** este campo de 8 bits indica el número de la sección contenida en la tabla, y se incrementa en "1" por cada sección adicional de la tabla.
- **last section number:** este campo de 8 bits indica el número de la última sección de la tabla.

2.3.3.5 Tablas

Las tablas son estructuras que sirven para transmitir información específica sobre la conformación del flujo TS, como por ejemplo los programas que están siendo transmitidos y cada uno de los flujos que conforman estos programas. Como ya se indicó anteriormente las tablas pueden ser privadas como las tablas SI (*System Information*) y DSM-CC y no privadas como las tablas PSI (*Program Specific Information*).

Una tabla puede ocupar mas de un paquete TS y cuando esto sucede todos los paquetes TS deben tener el mismo PID, de modo que el receptor pueda separarlos del flujo TS. En la Tabla 2.4 se detalla la composición de la tablas que fueron utilizadas en el desarrollo del software para analizar y generar el flujo BTS.

Tabla 2.4: Tablas PSI/SI.

Tabla		PID	Table ID
Nombre	Significado		
PAT	Program Association Table	0000 _H	00 _H
PMT	Program Map Table	Asignado por la PAT	02 _H
NIT	Network Information Table	0010 _H	40 _H y 41 _H

2.3.3.5.1 Tabla PAT

La tabla PAT informa los valores de los PID de las tablas PMT que conforman el flujo TS, existe una tabla PAT por cada flujo TS, la cual debe transmitirse cada 100 ms. En la Figura 2.10 se muestra la estructura de la tabla PAT.

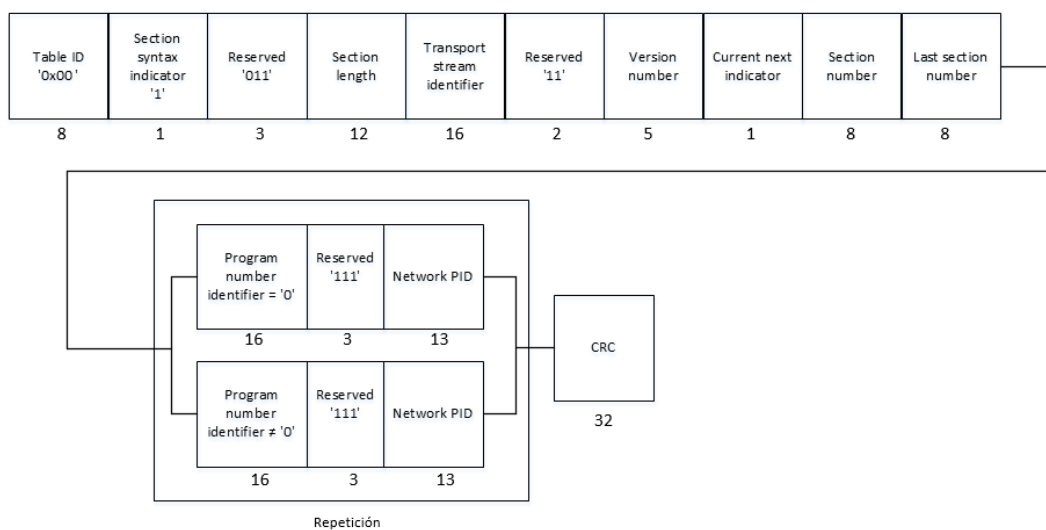


Figura 2.10: Tabla PAT

Cabe recalcar que los campos enmarcados con la palabra repetición de la Figura 2.10, se repetirán tantas veces dependiendo el número de programas transportados en el flujo TS. A continuación se describe la semántica de los campos de la tabla PAT, con excepción de los campos que ya fueron descritos en la Sección 2.3.3.4.1.

- ***program number ID***: este campo de 16 bits indica el número del programa al cual pertenece el el siguiente campo PMT PID, cuando se tiene el valor de 0x00 entonces el próximo PID obligatoriamente será el PID de la red, caso contrario el PID pertenecerá a uno de los programas del flujo TS.
- ***network PID***: este campo de 13 bits contiene el PID de los paquetes TS que deben contener la tabla NIT, y aparece únicamente cuando el valor del *program number* es igual a 0x00.
- **PMT PID**: Este campo de 13 bits contiene el PID de las tablas PMT que están siendo transmitidas en el flujo TS.
- **CRC**: Este campo de 32 bits contiene un código de redundancia cíclica que genera un salida igual a cero en los registros del decodificador, después de haber procesado toda una sección

2.3.3.5.2 Tabla PMT

La tabla PMT contiene información específica de los programas transportados en el flujo TS, como los tipos de flujos que contienen cada uno de estos programas con su respectivo PID, así como también el PID del *Program Clock Reference* (PCR) [13]. En el flujo TS son transportadas tantas tablas PMT como programas, es decir cada tabla contiene información de un solo programa. El período de transmisión de la tabla PMT para servicio fijo es 100 ms y para servicio móvil 200 ms. En la Figura 2.11 se muestra la estructura de la tabla PMT.

Los campos enmarcados en el cuadro con la palabra repetición de la Figura 2.11,

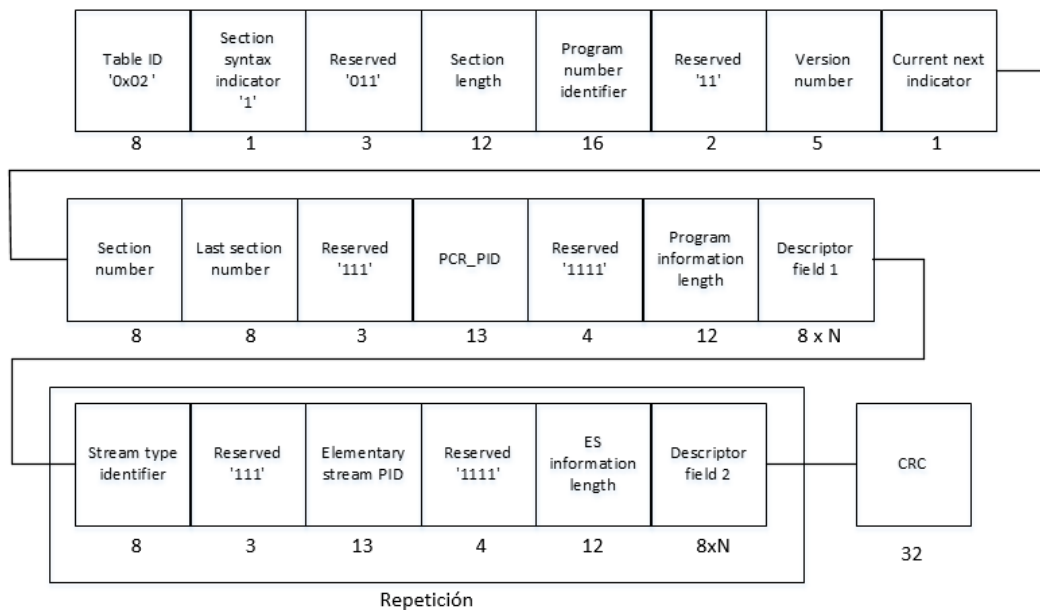


Figura 2.11: Tabla PMT

se repetirán de acuerdo al valor que tome el campo *program information length*. A continuación se describe la semántica de los campos de la tabla PMT, con excepción de los campos que ya fueron descritos en las Secciones 2.3.3.4.1 y 2.3.3.5.1.

- **PCR PID:** este campo de 13 bits indica el PID de los paquete TS que contienen el PCR Program Clock Reference, que es usado por el receptor en la decodificación de video, audio y datos.
- ***program information length:*** este campo de 12 bits indica el tamaño total en bytes de los descriptores en el caso que existieran, si no hay ningún descriptor que está siendo transmitido en la tabla PMT, este campo toma el valor de 0x00.
- ***stream type identifier:*** este campo de 8 bits contiene un identificador de los tipos de *streams* que están siendo transportados por cada uno de los programas, se repite de acuerdo al número de *streams* que contiene cada programa. Los tipos de flujos elementales que utiliza cada programa se muestran en la Tabla 2.5.

Tabla 2.5: Tipos de flujos elementales.

Decimal	Hexadecimal	Descripción
0	0x00	Reservado
1	0x01	ISO/IEC 11172-2 (MPEG-1 video)
2	0x02	Video según ITU H.262
3	0x03	ISO/IEC 11172-2 (MPEG-1 Audio)
4	0x04	Audio según ISO/IEC 12818-3
5	0x05	ITU-T Rec. H.222.0 ISO/IEC 13818/1 Sección Privada
6	0x06	Paquete PES con Información Privada
7	0x07	Cuadrícula de fondo de destino
8	0x08	Ventana de video
9	0x09	Sistema de acceso condicional y EMM / ECM PID
10	0x0A	ISO 639 idioma y tipo de audio
11	0x0B	Reloj del sistema de referencia externa
12	0x0C	Límites de utilización de búfer Multiplex
13	0x0D	Sistema de identificación de los derechos de autor y referencia
14	0x0E	Velocidad máxima de bits
15	0x0F	Indicador de datos privados
16	0x10	Smoothing buffer
17	0x11	STD video buffer leak control
18	0x12	Vídeo IBP indicador I-frame
19-127	0x13-0x7F	ITU-T Rec. H.222.0 ISO/IEC 13818/1 Reservado
128-255	0x80-0xFF	Uso Privado

- **elementary stream PID:** este campo de 13 bits contiene el PID de los paquetes que contienen cada uno de los *streams* de los programas.

- **ES information length:** este campo de 12 bits indica el tamaño de los descriptors de cada *stream*, si el *stream* no tiene ningún descriptor asignado este campo toma el valor de 0x00.

Dentro del campo *descriptor field 1* de la Figura 2.11 se incluyen algunos descriptors que proporcionan información adicional a la mencionada anteriormente; en especial para el desarrollo de este *software* se hace énfasis en el **descriptor de información de emergencia**, usado para verificar el estado del bit de la bandera de alarma de emergencia del campo "ISDB-Info" de cada paquete TSP. A continuación se hace una descripción mas detallada de este descriptor.

Descriptor de información de emergencia: Este descriptor es usado cuando las radiodifusoras transmiten alarmas de emergencia [14]. En la Figura 2.12 se muestra la estructura de este descriptor.

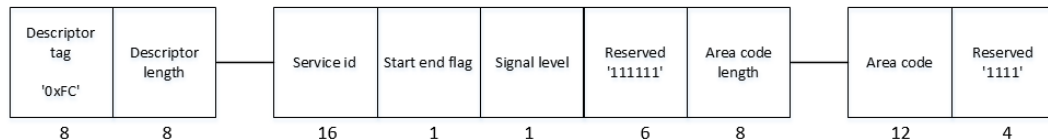


Figura 2.12: Descriptor de información de emergencia

La semántica de los campos de la Figura 2.12 se presentan a continuación.

- **service id:** este campo de 16 bits tiene el mismo valor del *program number* de la tabla PMT e indica el numero de evento transmitido.
- **start end flag:** este bit se activa durante toda la transmisión de la señal de emergencia y se desactiva cuando esta deja de ser transmitida.
- **signal level:** este bit indica el tipo de la señal de emergencia, cuando se activa indica una señal de segundo tipo, y cuando se desactiva indica una señal de primer tipo.

- **area code length:** este campo de 8 bits indica el tamaño en bytes del código de área.
- **area code:** este campo de 12 bits contiene el código de área asignado por los órganos responsables.

2.3.3.5.3 Tabla NIT

La tabla NIT contiene información de la organización física de la agrupación de multiplexores existentes en una misma red y sus características, así como todos los datos relevantes sobre la sintonía de los servicios existentes [13]. En la Figura 2.13 se muestra la estructura de la tabla NIT.

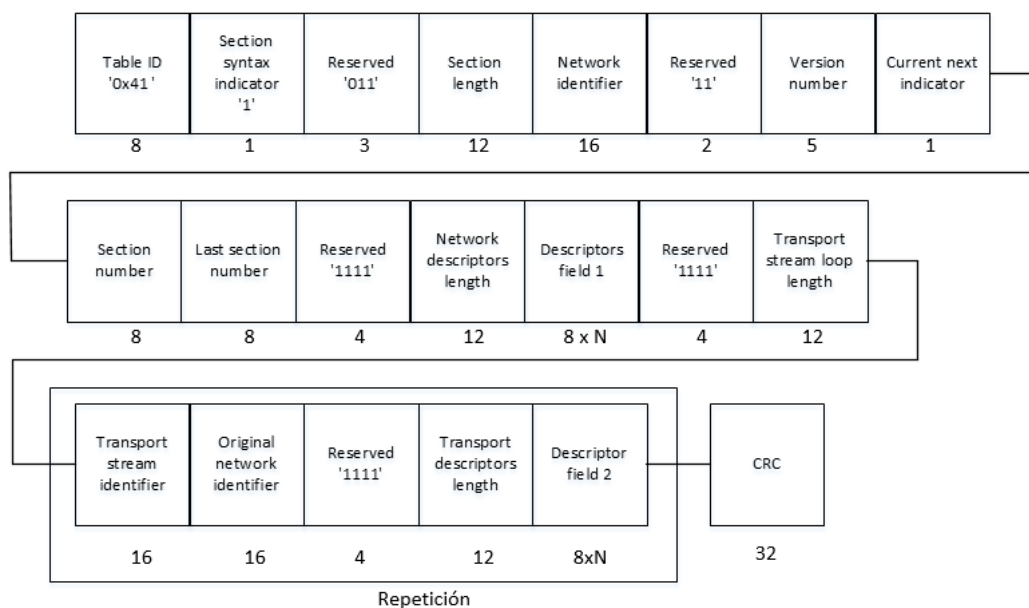


Figura 2.13: Tabla NIT

A continuación se describe la semántica de los campos de la tabla NIT, con excepción de los campos que ya fueron descritos en las Secciones 2.3.3.4.1 y 2.3.3.5.1.

- **network descriptor length:** este campo de 12 bits indica el tamaño total en bytes de los descriptores de red. En caso de no existir ningún descriptor de red este campo toma el valor de 0x00.

- ***descriptor field 1***: en este campo se insertan los descriptores de red, dependiendo el valor que haya tomado el campo *network descriptor length*.
- ***transport stream loop length***: este campo de 12 bits indica el tamaño total en bytes de los campos enmarcados en el recuadro con la palabra repetición, y va desde el primer *transport stream identifier* hasta antes del primer byte del campo CRC.
- ***transport stream identifier***: este campo de 16 bits permite identificar el flujo TS, de cualquier otro flujo del sistema de transmisión.
- ***original network identifier***: este campo de 16 bits tiene el mismo valor del *network id*.
- ***transport descriptors length***: este campo de 12 bits indica el tamaño total en bytes de los descriptores del *campo descriptor field 2*.
- ***descriptor field 2***: en este campo se insertan los descriptores del *loop* de TS.

Al igual que en la tabla PMT, en la tabla NIT se incluyen algunos descriptores en el campo *descriptor field 1* de la Figura 2.13. Uno de los descriptores utilizados para el desarrollo de este *software* fue el **descriptor de recepción parcial** el cual es descrito a continuación.

Descriptor de recepción parcial: Este descriptor permite la identificación del servicio conocido como one-seg, a continuación en la Figura 2.14 se muestra la estructura de este descriptor.

La semántica de los campos de la Figura 2.14 se presentan a continuación.

- ***service id***: este campo de 16 bits debe ser igual al campo *program number* en la sección correspondiente del mapa del programa, y debe indicar el *service id* del servicio *one-seg*.

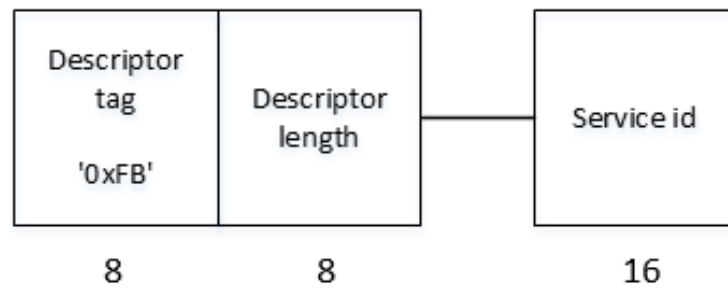


Figura 2.14: Descriptor de recepción parcial

2.3.4 Remultiplexor

Como ya se menciona en la sección 2.3.2, el estándar ISDB-Tb permite la transmisión de los paquetes hasta en tres capas jerárquicas, pero para que esto suceda es necesario añadir una etapa de remultiplexación luego del multiplexor de paquetes TS MPEG-2, el dispositivo incorporado en esta etapa es conocido como remultiplexor y su objetivo es combinar los paquetes asignados a cada capa jerárquica en un único flujo binario de salida conocido como **Flujo BTS**.

El flujo TS MPEG-2 no está preparado para realizar transmisiones en donde la información responda a un orden jerárquico y, menos aún, para permitir la recepción parcial [11].

Es por eso que el diseño del sistema ISDB-Tb adapta el flujo TS para que pueda cumplir con los dos requerimientos mencionados mediante el denominado remultiplexor, que además de las funciones nombradas anteriormente cumple con lo siguiente.

- Agrega 16 bytes al final de cada paquete TS, 8 corresponden al campo de "ISDBInfo" y los 8 restantes al bloque de paridad Reed Solomon, con lo que se forman nuevos paquetes de 204 bytes conocidos como TSP.
- Combina todos los paquetes de entrada, y entrega a su salida un único flujo binario con una tasa constante de 32,5 Mbps, lo cual se consigue insertando una

cierta cantidad de TSP nulos que dependerá de la configuración adoptada para cada capa jerárquica.

- Posiciona y dispone los paquetes TSP para posibilitar la transmisión jerárquica y la recepción parcial.

La asignación de los paquetes a cada una de las capas jerárquicas se lo hace mediante la extracción de la información del PID de cada uno de los paquetes, con lo cual se procede a asignar todos los paquetes con el mismo PID a la misma capa jerárquica.

2.3.5 Flujo de transporte para *broadcast* (BTS)

Para que la transmisión jerárquica sea posible se deben cumplir con las siguientes condiciones.

- Cada uno de los 13 segmentos mencionados en la sección 2.3.2.7 debe transmitir un número entero de paquetes TSP en un tiempo igual a la duración del cuadro OFDM.
- De la misma manera, cada una de las capas jerárquicas debe tener asignado un número entero de segmentos que dependerá de la configuración de transmisión.
- La frecuencia del reloj de sincronización del flujo BTS debe ser constante e igual a un múltiplo entero de la frecuencia de muestreo de la transformada inversa rápida de Fourier (IFFT).
- Los retardos introducidos entre las capas jerárquicas se compensan mediante un ordenamiento adecuado de los paquetes TSP dentro del cuadro multiplex [15].
- Los paquetes que contiene el *Program CLock Reference* y las tablas PSI deben ser transmitidos por la capa jerárquica más robusta.

2.3.5.1 Asignación del número de paquetes TSP

Un cuadro OFDM es una sucesión de símbolos OFDM, y su principal función es delimitar la secuencia de datos transmitidos para poder identificar el inicio y final de la transmisión. El tiempo de duración del cuadro OFDM está dado por la ecuación 2.2.

$$T_F = F \times T_S \quad (2.2)$$

Donde F y T_S son la cantidad de símbolos en el cuadro y la duración de cada símbolo respectivamente. Por otro lado se ha visto que la tasa de transmisión para los 13 segmentos esta dada por la ecuación 2.3.

$$R = K_O K_I \times \frac{b_P(13 \times L_D)}{T_S} \quad (2.3)$$

Donde:

- K_O es la tasa del código RS y su valor es 188/204.
- K_I , es la tasa del código convolucional, los valores que puede tomar son: 1/2, 2/3, 3/4, 5/6, 7/8.
- b_P es el número de bits requeridos para la modulación (QPSK = 2, 16QAM = 4, 64 QAM =6).
- L_D es el número de portadoras en cada segmento.

El número de bits de datos transmitidos durante la duración del cuadro OFDM por un segmento está dado por la ecuación 2.4.

$$b_T = R \times \frac{T_F}{13} \quad (2.4)$$

Cabe recalcar que de los 204 bytes de cada paquete TSP, solamente 188 bytes son de datos, por lo que el número total de bits de datos transmitidos por N paquetes TSP está dado por la ecuación 2.5.

$$b_T = (188 \times 8) \times N \quad (2.5)$$

Reemplazando la ecuación 2.3 en la ecuación 2.4, luego igualándola con la ecuación 2.5, y reemplazando los valores de $K_O = 188/204$ y $F = 204$ se obtiene la ecuación 2.6, que permite determinar el número de paquetes TSP que recibirá un segmento durante un cuadro OFDM de 204 símbolos.

$$N = \frac{K_1 \times b_P \times L_D}{8} \quad (2.6)$$

A continuación en la Tabla 2.6 se presentan los valores para N, de acuerdo a los modos, esquema de modulación y tasa de código convolucional.

Tabla 2.6: Valores de N por segmento.

<i>Modo</i>	Modulación		QPSK ($b_P = 2$)					16-QAM ($b_P = 4$)					64-QAM ($b_P = 6$)				
	K_I		1/2	2/3	3/4	5/6	7/8	1/2	2/3	3/4	5/6	7/8	1/2	2/3	3/4	5/6	7/8
1		96	12	16	18	20	21	24	32	36	40	42	36	48	54	60	63
2	L_D	192	24	32	36	40	42	48	64	72	80	84	72	96	108	120	126
3		384	48	64	72	80	84	96	128	144	160	168	144	192	216	240	252

2.3.5.2 Sincronización del BTS

La frecuencia del reloj f_r , permite establecer el ritmo de transmisión de bits durante el período T_F de un cuadro OFDM que está dado por la ecuación 2.7.

$$b_C = T_F \times f_R \quad (2.7)$$

Poniendo a T_F en función de T_U y a este en función de f_{IFFT} , se obtiene la ecuación 2.8.

$$b_C = 204 \times (1 + \Delta) \times \frac{2^P}{f_{IFFT}} \times f_R \quad (2.8)$$

Considerando los 204 bytes de cada TSP, se puede obtener la carga binaria total transportada por un número N_{BTS} de paquetes TSP para los 13 segmentos, que viene dada por la ecuación 2.9.

$$b_D = 204 \times 8 \times N_{BTS} \quad (2.9)$$

Igualando las ecuaciones 2.8 y 2.9 se obtiene:

$$\frac{f_R}{f_{IFFT}} = \frac{8 \times N_{BTS}}{2^P \times (1 + \Delta)} \quad (2.10)$$

Teniendo en cuenta que se puede asignar una cierta cantidad de segmentos a cada una de las capas jerárquicas, y además que el remultiplexor debe asignar una determinada cantidad de paquete nulos, para mantener la tasa binaria constante se tiene la ecuación 2.11.

$$N_{BTS} = N_A + q \times N_B + r \times N_C + \sum TSP_{Nulos} \quad (2.11)$$

Descartando los paquetes nulos de la ecuación 2.11 se tiene la siguiente desigualdad:

$$N_{BTS} > N_A + q \times N_B + r \times N_C \quad (2.12)$$

Asumiendo que los 13 segmentos son asignados a una misma capa jerárquica se debe cumplir que $N_{BTS} > 13 \times N$, por lo tanto la ecuación 2.10 se transforma en una desigualdad.

$$\frac{f_R}{f_{IFFT}} > \frac{8 \times 13 \times N}{2^P \times (1 + \Delta)} \quad (2.13)$$

La condición más desfavorable para la desigualdad se verifica para los valores más grandes de N ($b_P = 6yK_I = 7/8$) y el valor más pequeño de $\delta = 1/32$, y tomando el valor de $p = 11$ correspondiente al modo 1 se obtiene la frecuencia de reloj dada por la ecuación 2.14.

$$f_R = 4 \times f_{IFFT} \quad (2.14)$$

Reemplazando el valor de f_R en la ecuación 2.10, se obtiene la ecuación 2.15, necesaria para determinar el número NBTS de paquetes TSP que conforman el Cuadro Múltiplex.

$$N_{BTS} = 2^{p-1} \times (1 + \Delta) \quad (2.15)$$

El Cuadro Múltiplex es una sucesión de paquetes TSP con los que se construye un cuadro OFDM, y además tiene un tiempo de duración igual a la del cuadro OFDM [11]. En la Tabla 2.7 se muestra la cantidad N_{BTS} de paquetes TSP en un Cuadro Múltiplex para cada modo e intervalo de guarda.

Tabla 2.7: Cantidad N_{BTS} de paquetes TSP en un Cuadro Múltiplex.

Modo	L_D	Relación Δ			
		1/4	1/8	1/16	1/32
1	96	1280	1152	1088	1056
2	192	2560	2304	2176	2112
3	384	5120	4608	4352	4224

Comparando las Tablas 2.6 y 2.7, se observa que el número paquetes transportados por el BTS en un Cuadro Múltiplex es mucho mayor que el número de paquetes calculados para los 13 segmentos, que se obtiene de la sumatoria del producto de los valores de la Tabla 2.6 con el número de segmentos asignados a cada capa jerárquica. La diferencia de ambos valores la constituyen los TSP nulos que se obtiene de la ecuación 2.16.

$$TSP_{Nulos} = N_{BTS} - (N_A + q \times N_B + r \times N_C) \quad (2.16)$$

2.3.6 Estructura de paquetes TSP, flujo BTS y cuadro múltiplex

En la Figura 2.15 se observa la estructura de los paquetes TSP, su inserción dentro del flujo BTS, y el concepto de Cuadro Múltiplex.

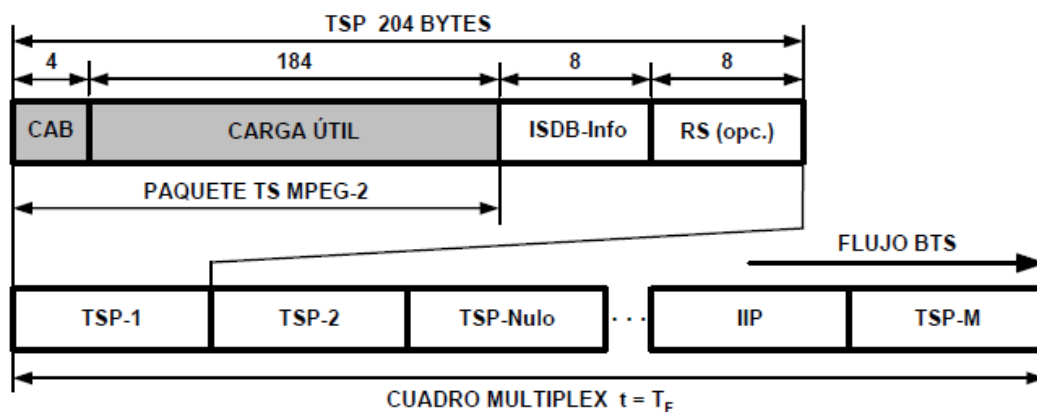


Figura 2.15: Estructura de los paquetes TSP, flujo BTS y cuadro múltiplex

Los paquetes TSP resultan de la adición de 16 bytes en la parte final de cada paquete TS, los cuales están organizados en dos campos, un campo conocido como "ISDB-Info" y un campo opcional conocido como "RS". El segundo campo mencionado contiene un código *Reed-Solomon* que permite corregir hasta 4 errores, y es un campo opcional ya que solamente es usado cuando el remultiplexor y el transmisor están físicamente separados [11].

Dentro del flujo BTS también se inserta un TSP especial conocido como IIP (*ISDB Information Packet*), que transporta entre otras cosas la información del TMCC (*Transmission and Multiplexing Configuration Control*), el cual a su vez contiene las señales de control necesarias para el correcto funcionamiento del receptor. En la Figura 2.15 se aprecia el concepto de Cuadro Múltiplex y Flujo BTS, aclarando que cada Cuadro Múltiplex contiene solamente un paquete IIP, el cual es insertado en reemplazo de uno de los paquetes nulos del Cuadro Múltiplex.

El IIP tiene dos descriptores:

- MCCI (Modulation Control Configuration Information)
 - Modo.
 - Intervalo de guarda.
 - Sincronización del TMCC.
 - Tipo de multiplexación de canal AC.
 - TMCC.

- NSI (Network Synchronization Information)
 - Referencia del tiempo de sincronización.
 - Tiempo máximo de retardo.
 - Código de identificación de los transmisores.
 - Configuración de retardos.

2.3.6.1 Campo ISDB-Info

Este campo constituye los primeros 8 bytes de los 16 que se añade a cada paquetes TS para formar los TSP de 204 bytes. La información que se incluye en cada uno de estos bytes se detalla en la Tabla 2.8.

La descripción de los campos de la Tabla 2.8 se presentan a continuación, donde el bit 0 es el menos significativo.

- **Identificador del TMCC:** Estos dos bits tomarán su valor de acuerdo a la Tabla 2.9.
- **Reservado:** Este bit siempre estará puesto en 1.
- **Bandera de reset de buffer de dispositivo sincronizado:** Este bit tendrá el valor de 0 en estado normal y 1 para reset.

Tabla 2.8: Estructura de la información del campo ISDB-Info.

Byte	Bits	Descripción
0	$b_7 - b_6$	Identificador del TMCC
	b_5	Reservado
	b_4	Bandera de reset de buffer de dispositivo sincronizado
	b_3	Control de inicio de alarma EBA
	b_2	Indicación del paquete de cabecera de cuadro con cambios en TMCC
	b_1	Señalización del TSP cabecera del cuadro múltiplex
	b_0	Señalización del sincronismo de cuadro OFDM
1	$b_7 - b_4$	Identificación de la capa jerárquica asignada al TSP
	$b_3 - b_0$	Indicador de cambio de parámetros de transmisión
2	b_7	Señalización de datos AC
	$b_6 - b_5$	Cantidad de bytes con datos AC
	$b_4 - b_0$	Contador de TSP
3	$b_7 - b_0$	
4	$b_7 - b_0$	Datos auxiliares para canal AC
5	$b_7 - b_0$	
6	$b_7 - b_0$	
7	$b_7 - b_0$	

- **Control de inicio de alarma EBA:** Este bit tendrá el valor de 0 en estado normal y 1 para indicar que la alarma de emergencia se está transmitiendo.
- **Indicación del paquete de cabecera de cuadro con cambios en TMCC:** Este bit tendrá el valor de 0 normalmente y 1 si la cuenta regresiva retorna a 1111.
- **Señalización del TSP cabecera del cuadro múltiplex:** Este bit tendrá el valor

Tabla 2.9: Estructura del identificador de TMCC.

Valor	Significado
00	Satélite
01	Reservado
10	Televisión Digital Terrestre
11	Radiodifusión Sonora Terrestre

de 1 para indicar que el TSP es cabecera del cuadro múltiplex y el valor de 0 para indicar que el TSP es uno de los restantes del cuadro múltiplex.

- **Señalización del sincronismo de cuadro OFDM:** Este bit tendrá el valor de 0 durante el cuadro par y 1 durante el cuadro impar.
- **Identificación de la capa jerárquica asignada al TSP:** Estos 4 bits tomarán su valor de acuerdo a la Tabla 2.10.

Tabla 2.10: Estructura del identificador de capa jerárquica.

Valor	Significado
0000	TSP Nulo
0001	TSP asignado a capa jerárquica A
0010	TSP asignado a capa jerárquica B
0011	TSP asignado a capa jerárquica C
0100	TSP con datos para canal AC
100	Paquete IIP
0101 - 0111	TSP definido por el radiodifusor
1001 - 1111	

- **Indicador de cambio de parámetros de transmisión:** Estos 4 bits permanecen en 1111 mientras no exista ningún cambio en los parámetros de transmisión, e

irán disminuyendo de acuerdo a la Tabla 2.11 cuando exista algún cambio en los parámetros de transmisión.

Tabla 2.11: Indicador de cambio de parámetros de transmisión.

Valor	Significado
0000	1 cuadro antes del cambio de parámetro
0001	2 cuadro antes del cambio de parámetro
0010	3 cuadro antes del cambio de parámetro
0011	4 cuadro antes del cambio de parámetro
0100	5 cuadro antes del cambio de parámetro
0101	6 cuadro antes del cambio de parámetro
0110	7 cuadro antes del cambio de parámetro
0111	8 cuadro antes del cambio de parámetro
1000	9 cuadro antes del cambio de parámetro
1001	10 cuadro antes del cambio de parámetro
1010	11 cuadro antes del cambio de parámetro
1011	12 cuadro antes del cambio de parámetro
1100	13 cuadro antes del cambio de parámetro
1101	14 cuadro antes del cambio de parámetro
1110	15 cuadro antes del cambio de parámetro
1111	Sin cambio de parámetro

- **Señalización de datos AC:** Este bit tendrá el valor de 0 cuando existan datos de canal AC, y 1 cuando no existan datos de canal AC.
- **Cantidad de bytes con datos AC:** Estos 2 bits tomarán su valor de acuerdo a la Tabla 2.12.
- **Contador de TSP:** Estos 13 bits constituyen un contador que se inicializa en 0 y

Tabla 2.12: Cantidad de datos AC.

Valor	Significado
00	1 byte de datos AC
01	2 bytes de datos AC
10	3 bytes de datos AC
11	4 bytes ó ausencia de datos AC

va incrementando de uno en uno con cada TSP del cuadro múltiplex, incluyendo los TSP nulos y el paquete IIP.

- **Datos auxiliares para canal AC:** Estos 4 bytes son usados para datos auxiliares del canal AC, pero no tienen ninguna validez si la señalización de datos AC esta en 1. Cuando este campo no es utilizado se insertan bits puestas en 1 como relleno, se elige este valor porque si se deja todos los bits en 0 no habría cambios de fase en la señal y AC se transformaría en un piloto continuo [16].

2.3.6.2 Paquete IIP

Como ya se menciono anteriormente cada cuadro múltiplex incluye un paquete IIP, el cual contiene información necesaria para definir parámetros de modulación, control y configuración del sistema. La información incluida dentro del paquete IIP se muestran en la Tabla 2.13.

Tabla 2.13: Estructura de la información del paquete IIP.

N° de Bits	Descripción	Valor y significado
8	Cabecera del paquete TS	47 _H
1		'0'
1		'1'
1		'0'
13		1FF0 _H
2		'00'
2		'01'
4		va de 0 a 15.
16	Puntero del paquete IIP	Indica el número de paquetes que hay entre el paquete IIP y el próximo cuadro múltiplex.
160	MCCI	Contiene la información de modulación, control y configuración.
8	Número de paquete sub-IIP	Se usa cuando el campo NSI es mayor a 159 bytes, caso contrario su valor es 00 _H , indicando que existe un solo paquete IIP.
8	Número de último paquete sub- IIP	Contiene la cantidad de paquetes IIP necesarios para transportar la información de NSI cuando esta excede los 159 bytes, caso contrario su valor es 00 _H .
8	Longitud de la información de sincronización de red SFN	Indica la longitud en bytes de la información NSI.
1272	Información de sincronización para la red SFN (NSI)	Contiene los parámetros de configuración de la red de frecuencia única (SFN).
n x 8	Bytes de relleno	Bytes rellenos con FF _H , cuando la información NSI es menor a 159 bytes.

La estructura del MCCI se presenta a continuación:

- 1 bit de control de la palabra de sincronización del TMCC, que toma su valores de acuerdo a la Tabla 2.14.

Tabla 2.14: Palabra de sincronismo.

Valor	Significado
0	Palabra de sincronismo S0=0011010111101110
0	Palabra de sincronismo S1=1100101000010001

- 1 bit de esquema de multiplexación de datos auxiliares para canal AC, que toma su valores de acuerdo a la Tabla 2.15.

Tabla 2.15: Multiplexación para datos auxiliares.

Valor	Significado
0	Multiplexación en IIP
0	Multiplexación en ISDB-Info

- 2 bits reservados para uso futuro.
- 4 bits que contienen el indicador de cambio de parámetros de transmisión, los valores que pueden tomar están de acuerdo a la Tabla 2.11.
- 2 bits que contienen el modo actual, los valores que pueden tomar están de acuerdo a la Tabla 2.16.
- 2 bits que contienen el intervalo de guarda actual, los valores que pueden tomar están de acuerdo a la Tabla 2.17.
- 2 bits que contienen el modo de próxima información, los valores que pueden tomar están de acuerdo a la Tabla 2.16.

Tabla 2.16: Modos del flujo BTS.

Valor	Modo
00	Reservado
01	Modo 1
10	Modo 2
11	Modo 3

Tabla 2.17: Intervalos de guarda del flujo BTS.

Valor	Intervalo de guarda
00	1/32
01	1/16
10	1/8
11	1/4

- 2 bits que contienen el intervalo de guarda de próxima información, los valores que pueden tomar están de acuerdo a la Tabla 2.17.
- 102 bits que contienen la información del TMCC, mostrada en la Tabla 2.18.
- 10 bits puestos a 1, reservados para usos futuros.
- 32 bits que contienen el CRC, usado para protección de errores.

La descripción de los campos de la Tabla 2.18 se presentan a continuación.

- **Identificación del sistema:** estos 2 bits toman su valor de acuerdo a la Tabla 2.19.

Tabla 2.18: Información del TMCC.

Bits	Descripción	
$b_{20} - b_{21}$	Identificación del sistema.	
$b_{22} - b_{25}$	Indicador de parámetros de transmisión.	
b_{26}	Control de inicio de alarma EBA.	
b_{27}	Información actual	Indicador de servicio de recepción parcial.
$b_{28} - b_{40}$		Parámetros de transmisión de la capa A
$b_{41} - b_{53}$		Parámetros de transmisión de la capa B
$b_{54} - b_{66}$		Parámetros de transmisión de la capa C
b_{67}	Próxima información	Indicador de servicio de recepción parcial.
$b_{68} - b_{80}$		Parámetros de transmisión de la capa A
$b_{81} - b_{93}$		Parámetros de transmisión de la capa B
$b_{94} - b_{106}$		Parámetros de transmisión de la capa C
$b_{107} - b_{109}$	Bits utilizados en $ISDBT_{SB}$	
$b_{110} - b_{121}$	Reservado para usos futuros	

Tabla 2.19: Campo de identificación del sistema.

Valor	Significado
00	$ISDB - T_b$
01	$ISDB - T_{SB}$
10 y 11	Reservados

- **Indicador de cambio de parámetros de transmisión:** los valores que pueden

tomar estos bits están de acuerdo a la Tabla 2.11.

- **Control de alarma EBA:** este bit tendrá el valor de 0 en estado normal y 1 para indicar que la alarma de emergencia se está transmitiendo.
- **Información actual**
 - **Indicador de servicio de recepción parcial:** toma el valor de 1 cuando existe el servicio de one-seg y 0 cuando este no existe.
 - **Parámetros de configuración de la capa A**
 - * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa A, de acuerdo a la Tabla 2.20.

Tabla 2.20: Esquema de modulación del TMCC.

Valor	Significado
000	DQPSK
001	QPSK
010	16 QAM
011	64 QAM
100	Reservado
101	Reservado
110	Reservado
111	Capa no utilizada

- * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa A, de acuerdo a la Tabla 2.21.

Tabla 2.21: Relación de codificación interna del TMCC.

Valor	Significado
000	1/2
001	2/3
010	3/4
011	5/6
100	7/8
101	Reservado
110	Reservado
111	Capa no utilizada

* **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo de la capa A, de acuerdo a la Tabla 2.22.

Tabla 2.22: Entrelazado de tiempo del TMCC.

Valor	Modo 1	Modo 2	Modo 3
000	0	0	0
001	4	2	1
010	8	4	2
011	16	8	4
100	Reservado	Reservado	Reservado
101	Reservado	Reservado	Reservado
110	Reservado	Reservado	Reservado
111	Capa no utilizada	Capa no utilizada	Capa no utilizada

* **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa A, de acuerdo a la Tabla 2.23.

Tabla 2.23: Número de segmentos en capa capa jerárquica del TMCC.

Valor	Significado	Valor	Significado
0000	Reservado	1000	8 segmentos
0001	1 segmento	1001	9 segmentos
0010	2 segmentos	1010	10 segmentos
0011	3 segmentos	1011	11 segmentos
0100	4 segmentos	1100	12 segmentos
0101	5 segmentos	1101	13 segmentos
0110	6 segmentos	1110	Reservado
0111	7 segmentos	1111	Capa no utilizada

– Parámetros de configuración de la capa B

- * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa B, de acuerdo a la Tabla 2.20.
- * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa B, de acuerdo a la Tabla 2.21.
- * **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo de la capa B, de acuerdo a la Tabla 2.22.
- * **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa B, de acuerdo a la Tabla 2.23.

– Parámetros de configuración de la capa C

- * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa C, de acuerdo a la Tabla 2.20.
- * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa C, de acuerdo a la Tabla 2.21.
- * **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo

de la capa C, de acuerdo a la Tabla 2.22.

- * **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa C, de acuerdo a la Tabla 2.23.

- **Próxima información**

- **Parámetros de configuración de la capa A**

- * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa A, de acuerdo a la Tabla 2.20.
 - * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa A, de acuerdo a la Tabla 2.21.
 - * **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo de la capa A, de acuerdo a la Tabla 2.22.
 - * **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa A, de acuerdo a la Tabla 2.23.

- **Parámetros de configuración de la capa B**

- * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa B, de acuerdo a la Tabla 2.20.
 - * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa B, de acuerdo a la Tabla 2.21.
 - * **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo de la capa B, de acuerdo a la Tabla 2.22.
 - * **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa B, de acuerdo a la Tabla 2.23.

- **Parámetros de configuración de la capa C**

- * **Esquema de modulación:** 3 bits que indican el esquema de modulación de la capa C, de acuerdo a la Tabla 2.20.

- * **Relación de codificación interna:** 3 bits que indican la relación de codificación interna de la capa C, de acuerdo a la Tabla 2.21.
- * **Entrelazado de tiempo:** 3 bits que indican el entrelazado de tiempo de la capa C, de acuerdo a la Tabla 2.22.
- * **Número de segmentos:** 4 bits que indican el número de segmentos asignados a la capa C, de acuerdo a la Tabla 2.23.

CAPÍTULO 3

DESARROLLO DEL SOFTWARE

3.1 Introducción

En este capítulo se explica el desarrollo del *software* conocido como *Analizer & Generator* BTS, el cual permite crear el flujo de transporte para broadcast de televisión digital terrestre, a partir de un archivo con extensión *ts*, el cual contiene el flujo multiplexado de los paquetes de audio, video y datos de los programas que se desea transmitir. *Analizer & Generator* BTS también permite analizar la información del flujo BTS, mostrando en pantalla la distribución de programas, información del paquete IIP, información del campo ISDB-Info y algunos otros parámetros relevantes del BTS.

El lenguaje de programación que se escogió para el desarrollo del *software* fue Java debido a los beneficios que presenta este frente a otros como son el desarrollo de aplicaciones que puedan ser ejecutadas en diversos sistemas operativos, la programación orientada a objetos y la facilidad de uso, además se usó el entorno de desarrollo integrado NetBeans IDE.

Analizer & Generator BTS cuenta con una interfaz gráfica de usuario amigable, que permite manipular la aplicación de una manera sencilla e interactiva, tomando en cuenta que los usuarios que manejen la aplicación deberán tener los conocimientos necesarios para la correcta interpretación y configuración de la información mostrada en pantalla.

3.2 Arquitectura de software

El objetivo principal del desarrollo de la arquitectura de *software* consiste en definir la estructura y las propiedades globales de la aplicación, teniendo así una visión general del sistema y dejando de lado el detalle de cada uno de los componentes para etapas posteriores de diseño [17].

En esta sección se definen los módulos que integrarán el sistema y las funciones que desempeñarán cada uno de ellos, también se especifica la interacción que existirá entre estos; teniendo así un modelo del sistema que satisfaga las necesidades del usuario y además sirva como guía de desarrollo del *software Analyzer & Generator* BTS.

3.2.1 Arquitectura del Analizador

A continuación se presenta el diagrama de bloques del Analizador detallando las relaciones que existen entre cada uno de los componentes y describiendo las funciones que desempeñan cada uno de estos.

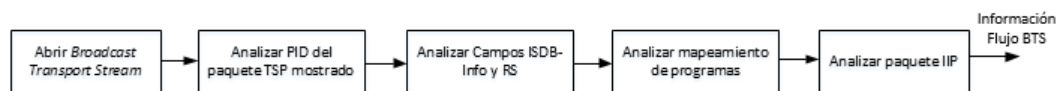


Figura 3.1: Diagrama de bloques del Analizador

La descripción de las funciones de cada bloque de la Figura 3.1 se lo hace de izquierda a derecha.

- **Bloque 1:** Selecciona la ubicación del archivo que contiene el *Broadcast Transport Stream*, muestra la información de los 204 bytes del primer paquete TSP. y la lista de programas contenidos en dicho flujo, cada uno con su PID correspondientes.

- **Bloque 2:** Toma el segundo y tercer byte del paquete TSP mostrado en pantalla y extrae la información de los que 13 bits que forman el PID, para luego mostrarla en pantalla.
- **Bloque 3:** Toma los 16 últimos bytes del paquete TSP mostrado en pantalla y extrae los Campos ISDB-Info y RS, para mostrar en pantalla el indicador de capa jerárquica, contador de TSP, señalización del TSP cabecera de cuadro, información de canales auxiliares, entre otros.
- **Bloque 4:** Busca dentro del *Broadcast Transport Stream*, la tabla PAT de la cual obtiene los PID y el *Program Number*, de cada programa transmitido en el flujo, con la información de los PID busca los paquetes TSP que contienen las tablas PMT, para seguidamente mostrar en pantalla la lista de programas y los tipos de flujos elementales que los conforman, cada uno de estos con sus respectivo PID.

Es necesario aclarar que el bloque marcado como 4 también es utilizado en la parte de generación, para brindar a los usuarios una mejor visualización de como realizar la asignación de capas jerárquicas de los paquetes TSP. Dicho bloque no es incluido en el diagrama del generador, ya que no constituye uno de los procesos de generación si no mas bien se lo utiliza para mejorar las prestaciones del *software*.

3.2.2 Arquitectura del Generador

A continuación se presenta el diagrama de bloques del Generador detallando las relaciones que existen entre cada uno de los componentes y describiendo las funciones que desempeñan cada uno de estos.

La descripción de las funciones de cada bloque de la Figura 3.2 se lo hace de izquierda a derecha.

- **Bloque 1:** Selecciona la ubicación del archivo que contiene el *Transport Stream*,

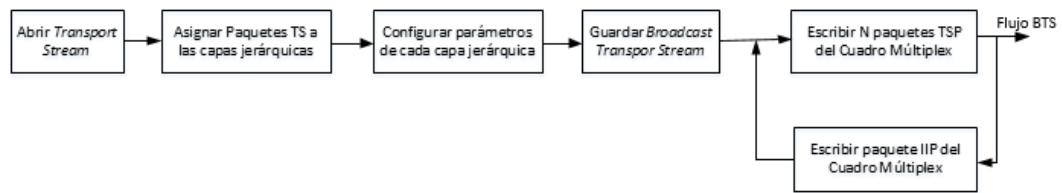


Figura 3.2: Diagrama de bloques del Generador

muestra la información de los 188 bytes del primer paquete TS y la lista de programas contenidos en dicho flujo, cada uno con su PID correspondientes.

- **Bloque 2:** Permite elegir la capa jerárquica por la que se enviarán cada uno de los paquetes que conforman el flujo BTS, la asignación se la hace de acuerdo al PID que tenga cada uno de los paquetes, obligando así al envío de todos los paquetes con el mismo PID por la misma capa jerárquica.
- **Bloque 3:** Configura la modulación y tasa de codificación interna de las 3 tres capas jerárquicas, además del modo e intervalo de guarda del flujo BTS. Todo estos parámetros serán usados para calcular el número de paquetes de cada Cuadro Múltiplex y de la posición del paquete IIP.
- **Bloque 4:** Verifica la asignación de los 13 segmentos a las capas jerárquicas, y comprueba que los parámetros de modulación y tasa de codificación de cada capa sean tal que las mismas soporten las tasas de transmisión asignadas de acuerdo a la asignación de paquetes elegida por el usuario. Luego de dichas verificaciones permite elegir la ubicación del archivo que contendrá el *Broadcast Transport Stream*.
- **Bloque 5:** Toma los datos del modo e intervalo de guarda configurados en el bloque 3 y calcula el número de paquetes TSP que debe tener cada Cuadro Múltiplex. Luego procede a formar los paquetes TSP que conforman el Cuadro Múltiplex, transcribiendo los 188 bytes de cada paquete TS del *Transport Stream*

y escribiendo los 16 bytes extras del campo ISDB-Info y RS.

- **Bloque 6:** Escribe el paquete conocido como IIP que contiene la información de modulación, control y configuración, de acuerdo a los parámetros configurados en el bloque 3. La posición del paquete IIP deberá ser al final de cada Cuadro Múltiplex.

Los bloques marcados como 5 y 6 se repetirán tantas veces sea necesario, dependiendo el tamaño del archivo que contiene el *Transport Stream* y del número de paquetes que tenga cada Cuadro Múltiplex.

3.3 Java

Como ya se dijo en la introducción de este capítulo el lenguaje escogido para la creación del *software* fue Java debido a las altas prestaciones que nos ofrece. Uno de los puntos más importantes a tratar en esta sección es el manejo de archivos con Java, ya que la parte esencial del software *Analizer & Generator BTS* se basa en la manipulación de los archivos que contienen el *Transport Stream* y *Broadcast Transport Stream*.

3.3.1 Archivos en Java

El objetivo general del *software* desarrollado, es la creación y análisis de los flujos de transporte para *broadcast*, los cuales deben estar contenidos en archivos que residan en alguna memoria y no desaparezcan cuando se termine de ejecutar la aplicación. Java ofrece algunas clases usadas para la creación y manipulación de archivos ubicadas dentro del paquete *java.io* las cuales se describen a continuación.

3.3.1.1 Clase File

Un objeto de la *clase File* representa el nombre de un archivo o directorio y permite obtener información de las propiedades del mismo. La *clase File* no permite escribir ni leer el contenido del archivo sino mas proporciona métodos para realizar operaciones a nivel de sistemas de archivos (listado de archivos, crear carpetas, borrar ficheros, cambiar nombre, etc) [18]. Cualquiera de los 3 constructores listados en la Tabla 3.1 pueden ser usados para crear un objeto de la clase File.

Tabla 3.1: Constructores de la clase File.

Constructor	Formato
public File(String pathname)	new File("C: \\ directorio \\ fichero.txt")
public File(File parent,String child)	new File("directorio","fichero.txt")
public File(File parent,String child)	new File(new File("directorio"),"fichero.txt")

Una vez creado el objeto de la clase File se puede obtener información del archivo usando cualquiera de los métodos mostrados en la Tabla 3.2

Tabla 3.2: Métodos de la clase File.

Método	Función
String getName()	devuelve el nombre del archivo
String getPath()	devuelve la ruta del archivo
boolean canRead()	indica si el archivo puede ser leído
boolean canWrite()	indica si el archivo puede ser escrito
long length()	devuelve el tamaño del archivo en bytes
void delete()	borra el archivo asociado al File
boolean exists()	indica si el archivo existe
void renameTo(File nvonombre)	renombra el archivo

3.3.1.2 Clase *RandomAccessFile*

Un objeto de la clase *RandomAccessFile* se utiliza para acceder a un archivo de forma aleatoria, permitiendo mover el puntero del archivo a cualquier posición dentro del mismo. La clase *RandomAccessFile* puede ser usada tanto para leer como para escribir bytes dentro de un archivo, en la posición indicada por el puntero la cual viene dada en bytes [19]. Cualquiera de los 2 constructores listados en la Tabla 3.3 pueden ser usados para crear un objeto de la clase *RandomAccessFile*.

Tabla 3.3: Constructores de la clase *RandomAccessFile*

Constructor
<code>public RandomAccessFile(String name,String mode),throws FileNotFoundException</code>
<code>public RandomAccessFile(File file,String mode),throws FileNotFoundException</code>

El modo citado en los 2 constructores de la Tabla 3.3 pueden ser:

- **"r"**: abre el archivo en modo lectura.
- **"rw"**: abre el archivo en modo lectura/escritura, en el caso de no exista el archivo lo crea ese momento.

La clase *RandomAccessFile* implementa las interfaces *DataInput* y *DataOutput*, por lo que tiene acceso a todas las operaciones *read()* y *write()* de las clases *DataInputStream* y *DataOutputStream* que también implementan las interfaces *DataInput* y *DataOutput*. Los métodos mas importantes de la clase *RandomAccessFile* son los mostrados en la Tabla 3.4

3.4 *Analizer & Generator BTS*

En esta sección se presenta la implementación del *software Analizer & Generator* BTS, para lo cual se realizarán los diagramas de flujo correspondientes al desarrollo del

Tabla 3.4: Métodos de la clase *RandomAccessFile*.

Método	Función
long getFilePointer()	retorna la posición del puntero del archivo
void seek(long posicion)	coloca el puntero del archivo en una posición determinada
long length()	retorna el tamaño del archivo en bytes
int skipBytes(int desplazamiento)	mueve el puntero desde la posición actual el número de bytes indicados

analizador y generador del *Broadcast Transport Stream*, además se hará una breve descripción de las clases y métodos usados para el desarrollo del *software*, por último se presentará el diseño de las interfaces gráficas de usuario.

3.4.1 Estructura del *Analyzer & Generator BTS*

La programación orientada a objetos fue una de las principales razones por las cuales se eligió Java para el desarrollo de la aplicación, ya que en la realidad uno de los principios fundamentales con los que se modela la creación de un programa bajo este concepto, es el principio de la modularidad, mediante el cual se propone al programador dividir su programa en módulos diferentes, cada cual con su propio sentido. Dicho concepto también permite asociar la programación de la aplicación con la arquitectura de *software* planteada en la Sección 3.2, en la cuál se definieron los módulos que integrarán el sistema y las funciones que desempeñarán cada uno de ellos.

El patrón MVC (Modelo-Vista-Controlador) fue utilizado para el diseño de la arquitectura del software, ya que de esta manera es posible separar los datos y la lógica del negocio de la aplicación de la interfaz de usuario, permitiendo facilitar el desarrollo de la aplicación y la modificación de la misma al utilizar las ideas de reutilización

de código y separación de conceptos [20]. Los componentes del patrón MVC son los siguientes:

- **Modelo:** Contiene el núcleo de funcionalidad de la aplicación, es decir incluye todas las clases vinculadas con la creación, lectura, actualización y eliminación de los datos que definen la lógica de la aplicación.
- **Vista:** Se encarga de interactuar con el usuario, presentado el 'modelo' en un formato adecuado para interactuar.
- **Controlador:** Comunica los objetos del modelo y la vista, es decir reacciona a las peticiones hechas por el usuario y ejecuta la tarea más adecuada para cumplir dicho requerimiento. Además se encarga de realizar la carga del procesamiento de datos de la aplicación.

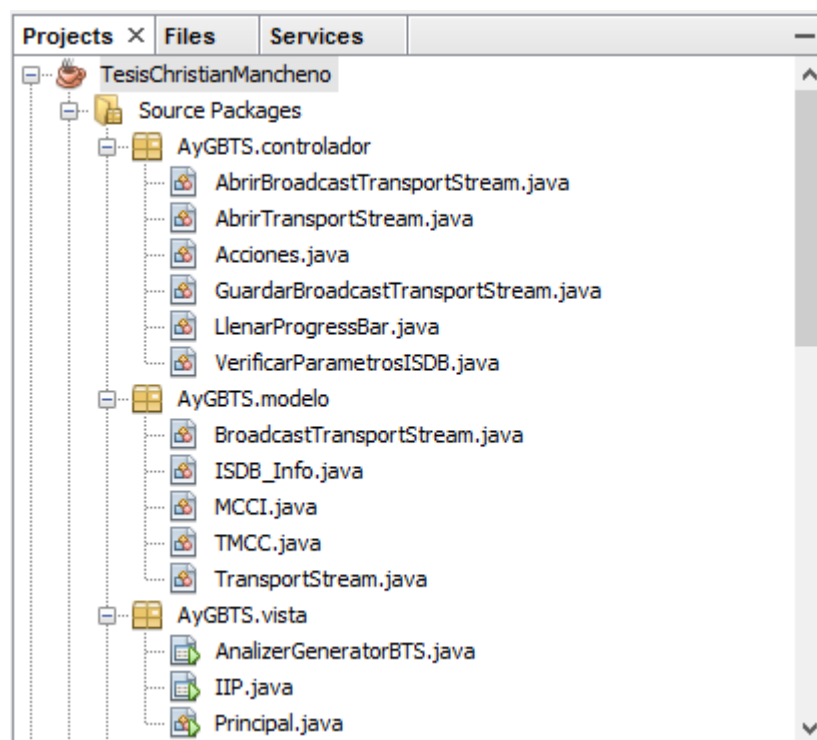


Figura 3.3: Uso del patrón MVC en el desarrollo de la aplicación.

La Figura 3.3 muestra el uso del patrón MVC en el desarrollo del *software Analyzer & Generator* BTS. Además se observa que cada paquete de la aplicación contiene un determinado número de clases. Las funciones que cumplen cada una de estas clases deberán estar de acuerdo al patrón de diseño MVC.

3.4.1.1 Paquete Controlador

Contiene 6 clases cuyas funciones son descritas a continuación.

3.4.1.1.1 Clase AbrirBroadcastTransportStream

Esta clase permite abrir el archivo que contiene el *Broadcast Transport Stream*, validando si efectivamente el archivo seleccionado contiene el flujo BTS, caso contrario despliega un mensaje de error indicando que el archivo no tiene el formato requerido. Luego de abrir el archivo esta clase despliega el contenido de los 204 bytes del primer TSP, y también muestra una lista de los programas transportados en el *Broadcast Transport Stream* con los respectivos flujos de audio, video y datos que contiene cada uno. La Figura 3.4 muestra el diagrama UML de la clase descrita.

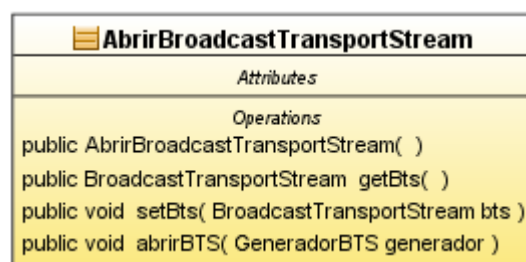


Figura 3.4: Diagrama UML clase AbrirBroadcastTransportStream.

3.4.1.1.2 Clase AbrirTransportStream

Esta clase permite abrir el archivo que contiene el *Transport Stream* que será usado de base para la generación del BTS, validando si el archivo seleccionado contiene el flujo TS, caso contrario despliega un mensaje de error indicando que el archivo no tiene el formato requerido. Luego de abrir el archivo esta clase despliega el contenido de

los 188 bytes del primer paquete TS, y también muestra una lista de los programas contenidos en el *Transport Stream* con los respectivos flujos de audio, video y datos que contiene cada uno. Además esta clase es la responsable de obtener la tasa de bits y el PID de cada de las tablas y los flujos elementales del *Transport Stream*. La Figura 3.5 muestra el diagrama UML de la clase descrita.

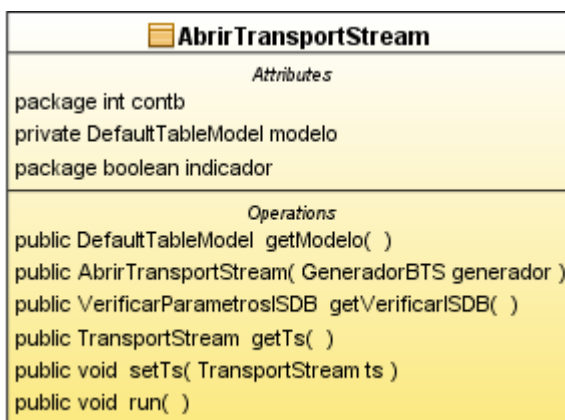


Figura 3.5: Diagrama UML clase AbrirTransportStream.

3.4.1.1.3 Clase Acciones

Esta clase contiene todos los métodos necesarios para la generación y el análisis del *Broadcast Transport Stream*. Es la clase que se encarga de casi todo el procesamiento dentro del *software*, por dicha razón se constituye en la más importante de todas ya que la mayoría de las clases de los 3 paquetes instancian un objeto de la clase Acciones. El diagrama UML de esta clase no ha sido incluido, debido a que es demasiado extenso.

3.4.1.1.4 Clase GuardarBroadcastTransportStream

Esta clase permite escribir en un archivo los bytes que formarán el *Broadcast Transport Stream*. El algoritmo usado para escribir el *Broadcast Transport Stream* es el siguiente: se transcribe los bytes del archivo que contiene *Transport Stream* y cada 188 bytes escritos se insertan los 16 bytes correspondientes a los Campos ISDB-Info y RS, luego de esto dependiendo del modo e intervalo de guarda elegido por el usuario, se inserta

el paquete IIP cada cierto número de paquetes TSP. Si la posición que debe ocupar el paquete IIP corresponde a la posición de un TSP nulo, se inserta el paquete IIP y el TSP nulo es eliminado, caso contrario si la posición corresponde a un TSP de datos, el IIP es insertado y el TSP de datos es desplazado a la posición del siguiente TSP nulo, el cual es eliminado. La Figura 3.6 muestra el diagrama UML de la clase descrita.

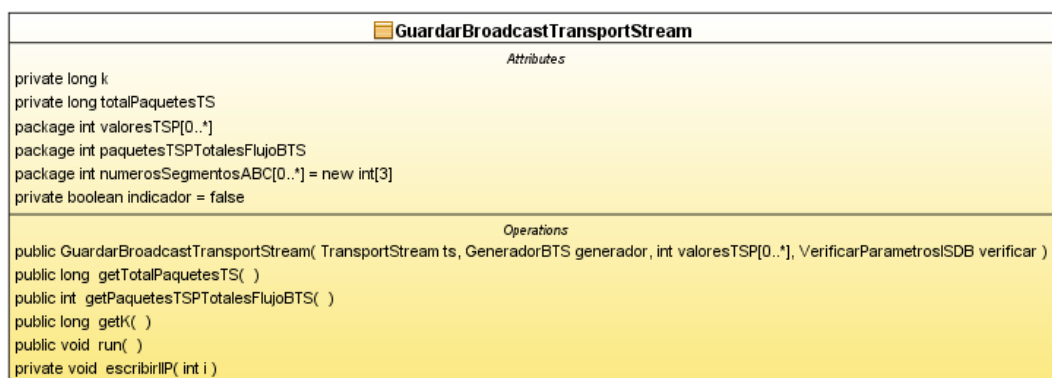


Figura 3.6: Diagrama UML clase GuardarBroadcastTransportStream

3.4.1.1.5 Clase LlenarProgressBar

Esta clase extiende la clase *Thread*, y se ejecuta concurrentemente con la clase *GuardarBroadcastTransportStream* mientras esta se encuentra escribiendo el flujo BTS en un archivo seleccionado. La principal función de esta clase es setear el contenido del *JProgressBar* que indica el porcentaje de información que ha sido almacenada. La Figura 3.7 muestra el diagrama UML de la clase descrita.

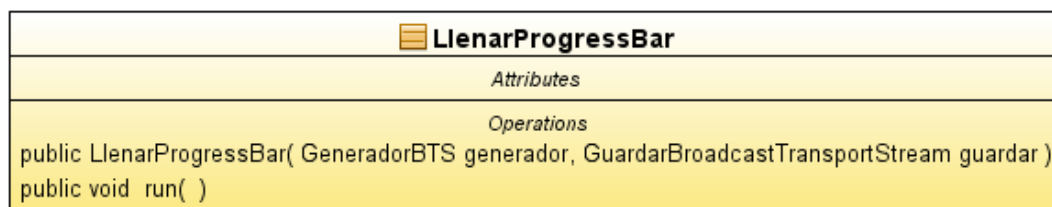


Figura 3.7: Diagrama UML clase LlenarProgressBar

3.4.1.1.6 Clase VerificarParametrosISDB

Esta clase extiende la clase *Thread*, y se ejecuta concurrentemente con la clase *AbrirTransportStream*. La principal función de esta clase es verificar si la suma del número de segmentos seleccionados en cada capa jerárquica no sobrepasa los 13 y también verificar si la tasa de bits de los PID seleccionados para cada capa jerárquica no exceda la capacidad máxima configurada para cada una de ellas según la modulación, intervalo de guarda y tasa de codificación interna elegida por el usuario para cada capa. La Figura 3.8 muestra el diagrama UML de la clase descrita.

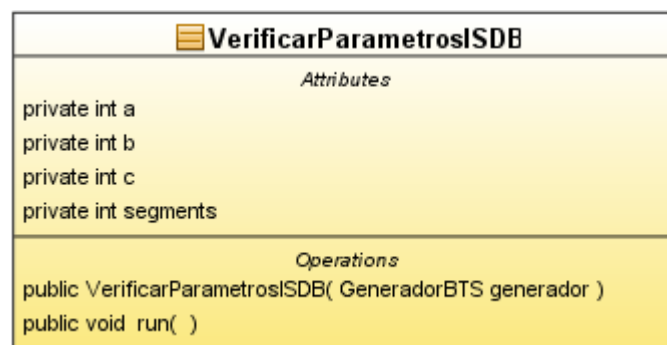


Figura 3.8: Diagrama UML clase VerificarParametrosISDB

3.4.1.2 Paquete Modelo

Contiene 5 clases cuyas funciones son descritas a continuación.

3.4.1.2.1 Clase *BroadcastTransportStream*

Esta clase contiene un objeto de la clase *RandomAccessFile*, el cual permite acceder de forma aleatoria a un archivo seleccionado por el usuario y escribir sobre el los bytes que conformarán el *Broadcast Transport Stream*. Además contiene un arreglo de tipo byte en el cual se almacenan los 204 bytes de cada paquete TSP que van hacer escritos en el archivo seleccionado. Esta clase posee los métodos `get` y `set` que permiten obtener y modificar los 2 atributos mencionados anteriormente. La Figura 3.9 muestra el diagrama UML de la clase descrita.

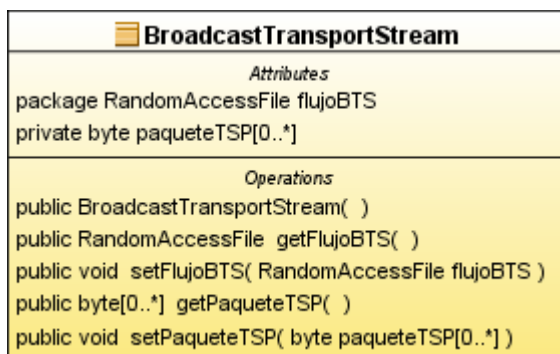


Figura 3.9: Diagrama UML clase BroadcastTransportStream

3.4.1.2.2 Clase MCCI

Esta clase contiene los arreglos donde se almacenan los bytes que conforman los primeros campos del descriptor MCCI, los cuales posteriormente serán escritos cada dentro del paquete IIP. Esta clase posee los métodos get de todos sus atributos que sirven para obtener cada uno cuando estos quieran ser escritos en el archivo. La Figura 3.10 muestra el diagrama UML de la clase descrita.

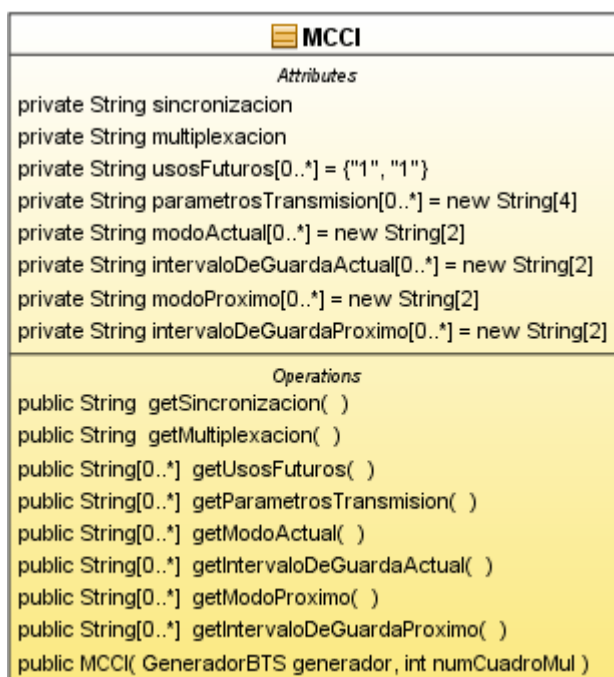


Figura 3.10: Diagrama UML clase MCCI

3.4.1.2.3 Clase TMCC

Esta clase contiene los arreglos donde se almacenan los bytes que conforman la información del TMCC, los cuales posteriormente serán escritos cada dentro del paquete IIP de acuerdo a las configuraciones realizadas para cada capa jerárquica. Esta clase posee los métodos get de todos sus atributos que sirven para obtener cada uno cuando estos quieran ser escritos en el archivo. La Figura 3.11 muestra el diagrama UML de la clase descrita.



Figura 3.11: Diagrama UML clase TMCC

3.4.1.2.4 Clase ISDBInfo

Esta clase contiene los arreglos donde se almacenan los bytes que conforman el campo ISDB-Info, los cuales posteriormente serán escritos cada 188 bytes en el archivo que contendrá el *BroadcastTransportStream*. Esta clase posee los métodos get de todos

sus atributos que sirven para obtener cada uno cuando estos quieran ser escritos en el archivo. La Figura 3.12 muestra el diagrama UML de la clase descrita.



Figura 3.12: Diagrama UML clase ISDBInfo

3.4.1.2.5 Clase *TransportStream*

Esta clase contiene un objeto de la clase *RandomAccessFile*, el cual permite acceder de forma aleatoria al archivo que contiene el *Transport Stream* para leer el contenido del mismo y almacenar la información obtenida en paquetes de 188 bytes, dentro de un arreglo de tipo `byte[188]` declarado dentro de esta clase. Esta clase posee los métodos *get* y *set* que permiten obtener y modificar los 2 atributos mencionados anteriormente.

La Figura 3.13 muestra el diagrama UML de la clase descrita.

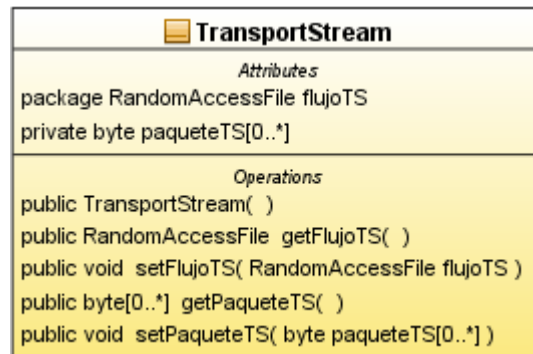


Figura 3.13: Diagrama UML clase TransportStream

3.4.1.3 Paquete Vista

Contiene 3 clases cuyas funciones son descritas a continuación.

3.4.1.3.1 Clase *AnalizerGeneratorBTS*

Extiende a la clase *JFrame*, sobre la cual se desarrolla la interfaz gráfica de usuario del analizador y generador de *Broadcast Transport Stream*. Esta clase contiene los métodos *get* y *set* de todos los componentes de las ventanas, los cuales permiten acceder a los mismos desde otras clases y modificar los datos que serán visualizados por el usuario. Las Figuras 3.14 y 3.15 muestran la interfaz del Analizador y Generador de BTS respectivamente.

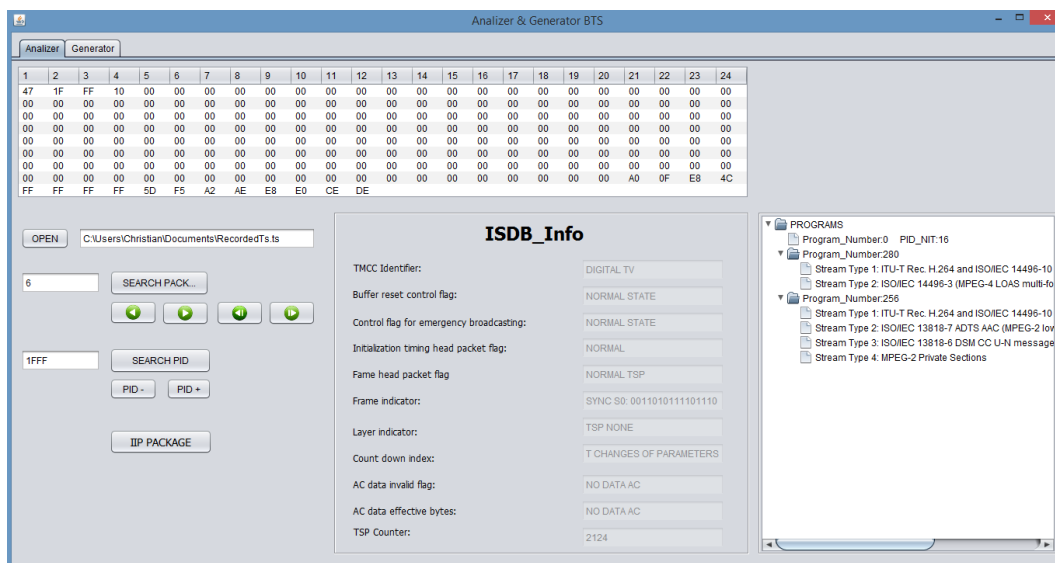


Figura 3.14: Interfaz gráfica de usuario del Analizador

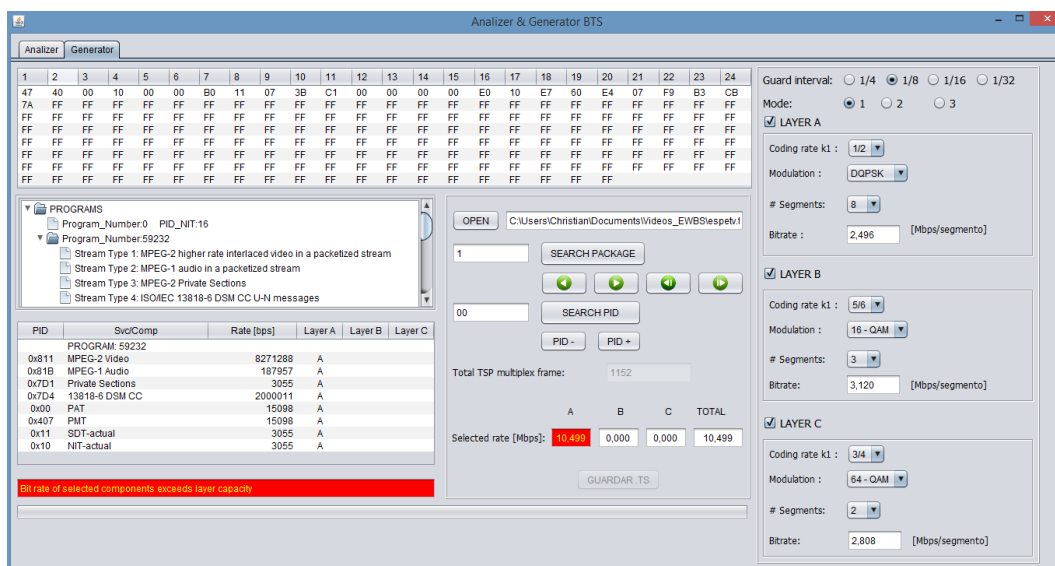


Figura 3.15: Interfaz gráfica de usuario del Generador

3.4.1.3.2 Clase IIP

Extiende a la clase *JFrame*, sobre la cual se desarrolla la interfaz gráfica de usuario donde se despliega la información del paquete IIP, la ventana se despliega al presionar el botón ubicado en la interfaz del Analizador llamado **IIP PACKAGE**. Esta clase

contiene los métodos *get* y *set* de todos los componentes de las ventanas, los cuales permiten acceder a los mismos desde otras clases y modificar los datos que serán visualizados por el usuario. La Figura 3.16 muestra la interfaz gráfica del paquete IIP.

The screenshot shows a graphical user interface for the 'IIP PACKAGE'. The window title is 'INFORMATION IIP PACKAGE'. The main title is 'IIP PACKAGE'. The interface includes several input fields for parameters:

- TMCC synchronization word: S0: 1100101000010001
- AC data effective position: Multiplexing in ISDB-Info
- Initialization timing indicator: Without changes of parameters

There are two columns for 'Current information' and 'Next information' for 'Mode' and 'Guard interval':

Parameter	Current information	Next information
Mode:	Mode 3	Mode 3
Guard interval:	1/16	1/16

The 'TMCC Information' section includes:

- System ID: ISDB-Tb
- Alert broadcasting: Normal State
- Count down index: ut changes of parameters

Below this are tabs for 'Current Information' and 'Next Information'. A 'Partial reception flag' is set to 'Yes'.

At the bottom, there is a table with columns 'Layer A', 'Layer B', and 'Layer C' for 'Modulation', 'Coding rate', 'Time interleaving', and 'Segments number':

Parameter	Layer A	Layer B	Layer C
Modulation:	QPSK	64 QAM	Layer not used
Coding rate:	2/3	3/4	Layer not used
Time interleaving:	2	2	Layer not used
Segments number:	1 segments	12 segments	Layer not used

A 'RETURN' button is located in the bottom right corner.

Figura 3.16: Interfaz gráfica del paquete IIP

3.4.1.3.3 Clase Principal

Esta clase contiene el método *main* que es invocado al ejecutar el programa, además instancia un objeto de la clase *AnalyzerGeneratorBTS* que permite visualizar la interfaz

gráfica de usuario de la aplicación. La Figura 3.17 muestra el diagrama UML de la clase descrita.

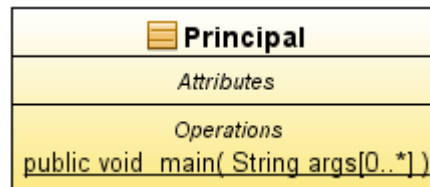


Figura 3.17: Diagrama UML clase Principal

3.4.2 Diagramas de flujo

Todas las clases y métodos descritos en la Sección 3.4.1, ayudan en el análisis y generación del *Broadcast Transport Stream*, pero para que cada uno de ellos funcione correctamente y cumpla la función que le ha sido asignada, deben seguir algoritmos o procesos que guíen su comportamiento. Los diagramas de flujos de los algoritmos usados en el análisis y generación del *Broadcast Transport Stream* son presentados a continuación.

Los algoritmos usados en la generación del BTS han sido representados en 3 diagramas de flujo descritos a continuación.

- **Primero:** La Figura 3.18 muestra el diagrama de flujo que permite obtener la lista de los programas con sus respectivos flujos de audio, video y datos que contiene el *Transport Stream*, y mostrarlos en el *JTree* de la pestaña del Generador.
- **Segundo:** La Figura 3.19 muestra el diagrama de flujo que permite llenar la tabla que contendrá todas las tablas, paquetes PES, PCR, etc, contenidos en el *Transport Stream*, para poder asignar por que capa serán enviados cada uno de estos.

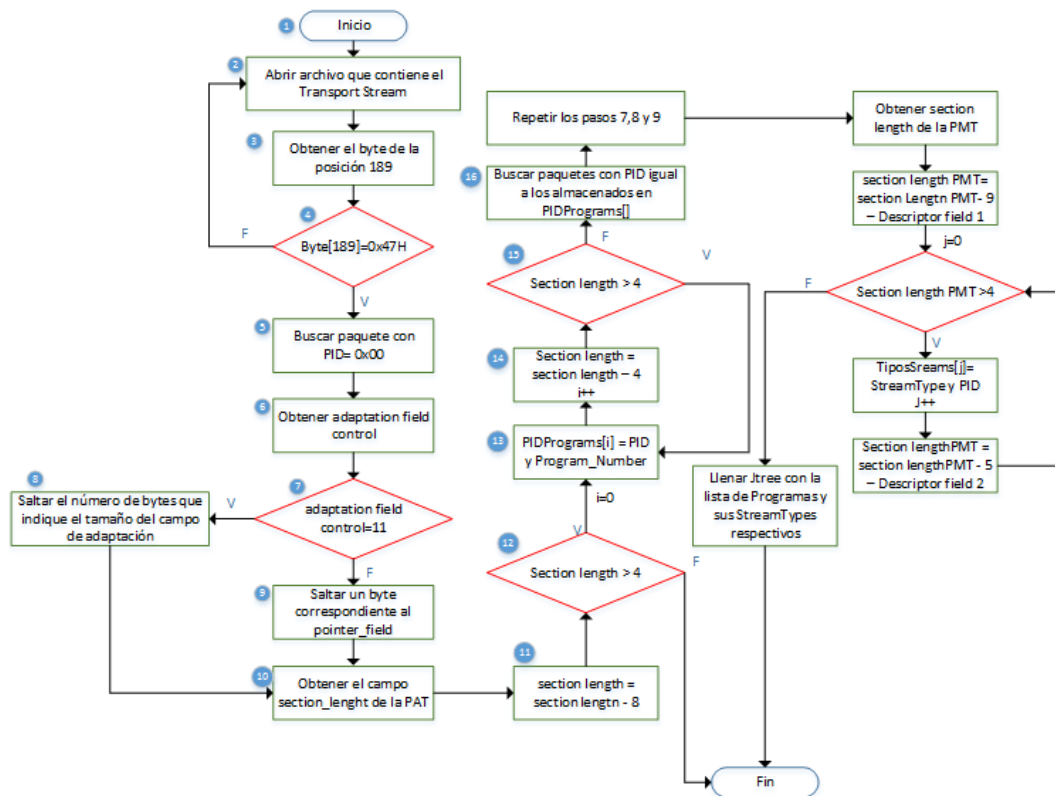


Figura 3.18: Diagrama de flujo del análisis de la PAT y PMT

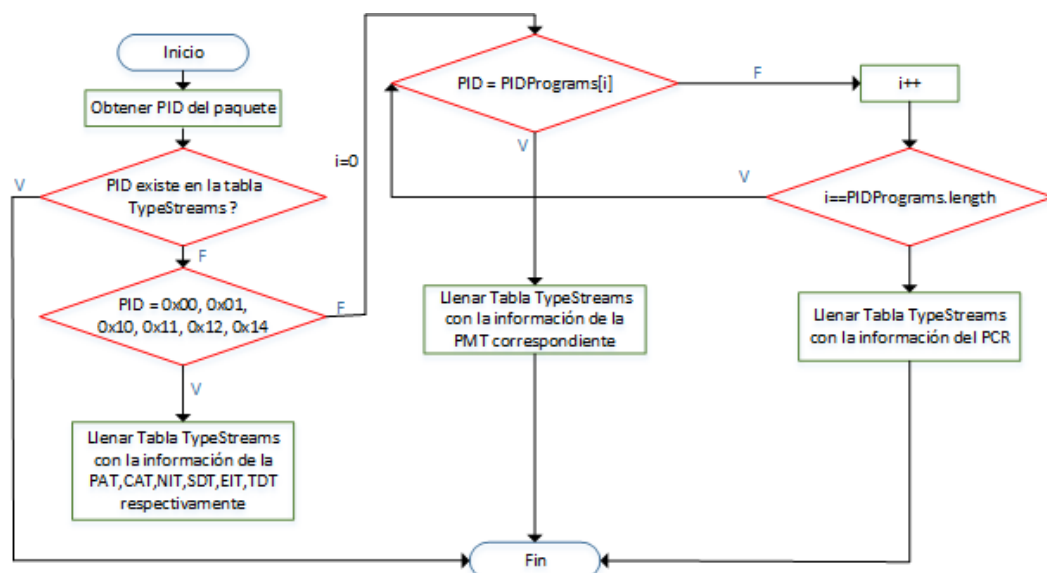


Figura 3.19: Diagrama de flujo para la obtención de todos los PID

- **Tercero** La Figura 3.20 muestra el diagrama de flujo del proceso general de almacenamiento de los bytes que forman el *Broadcast Transport Stream*.

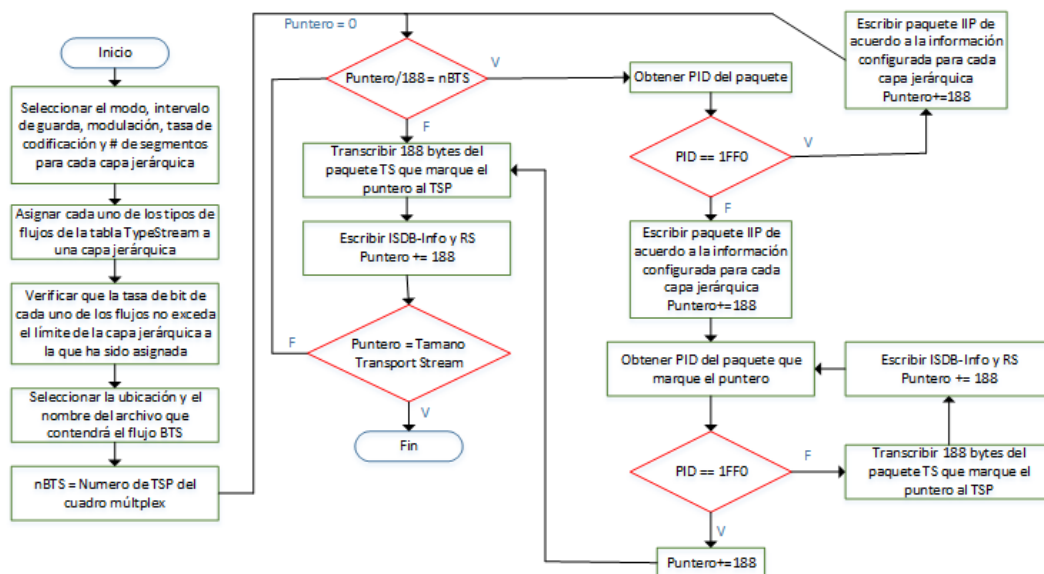


Figura 3.20: Diagrama de flujo del proceso de generación

Los algoritmos usados en el análisis del BTS han sido representados en 8 diagramas de flujo mostrados en 4 figuras y descritos a continuación.

- **Primero:** La Figura 3.18 muestra el diagrama de flujo que permite obtener la lista de los programas con sus respectivos flujos de audio, video y datos que contiene el *Broadcast Transport Stream*, los cambios que existirán para este diagrama serán: en el paso 2 se abrirá el archivo que contiene el *Broadcast Transport Stream*, y en el paso 3 se verificara que `byte[205]=0x47`.
- **Segundo:** La Figura 3.21 muestra los diagramas de flujo usados para la búsqueda de paquetes TSP por la posición que ocupan dentro del BTS y para el cambio de dichos paquetes hacia adelante y atrás.
- **Tercero:** La Figura 3.22 muestra los diagramas de flujo usados para la búsqueda de paquetes TSP por número de PID y para el cambio de los paquetes con el mismo PID hacia adelante y atrás.

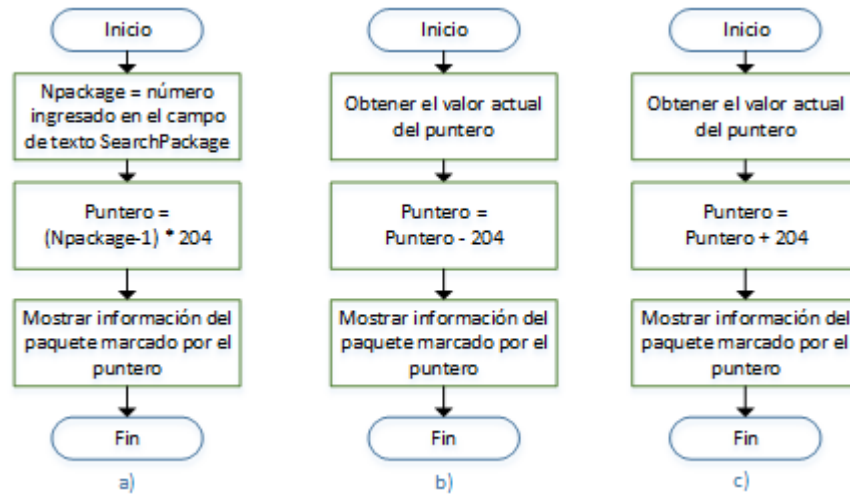


Figura 3.21: Diagramas de flujo del proceso de búsqueda de paquetes TSP

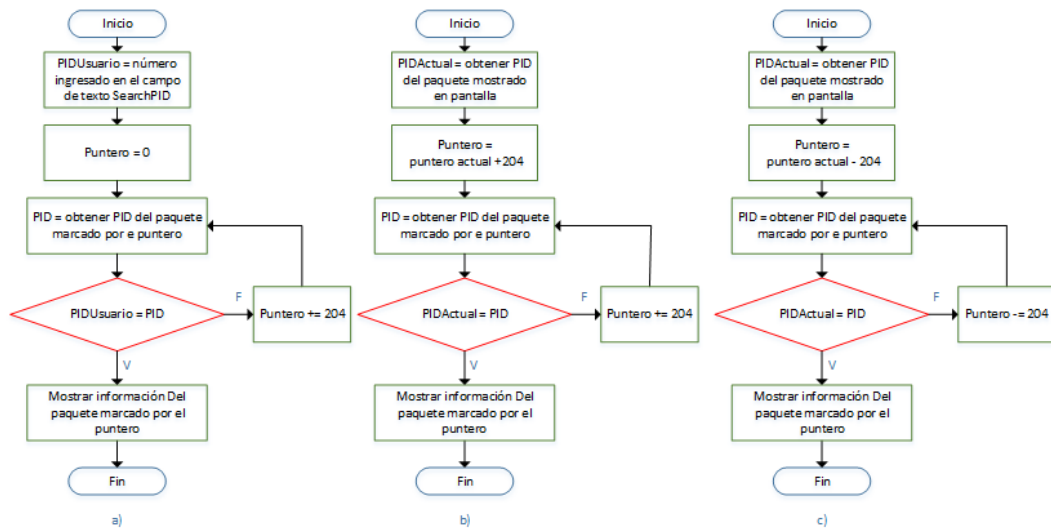


Figura 3.22: Diagramas de flujo del proceso de búsqueda de paquetes TSP por PID

- **Cuarto:** La Figura 3.23 muestra el diagrama de flujo usado para buscar el paquete IIP más cercano al paquete TSP mostrado en pantalla y desplegar su información.

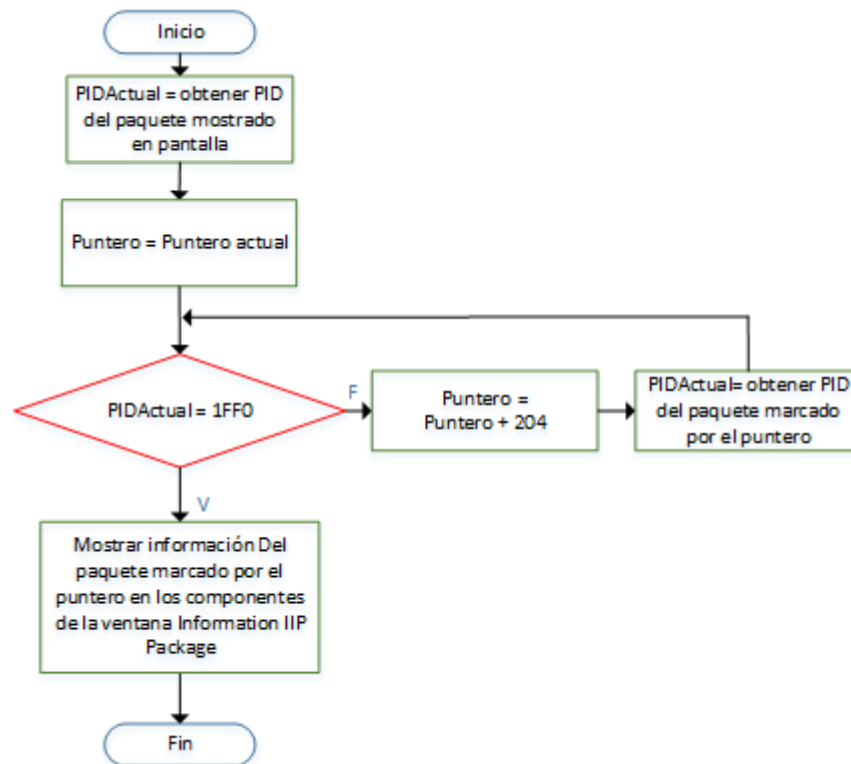


Figura 3.23: Diagramas de flujo del proceso de búsqueda de la información IIP

CAPÍTULO 4

PRUEBAS Y RESULTADOS

Una vez realizado el desarrollo del *software Analyzer & Generator BTS*, se procede a realizar pruebas de generación, análisis y transmisión del *Broadcast Transport Stream*. Los detalles de cada una de estas pruebas son presentadas a continuación.

4.1 Pruebas de generación

Las pruebas de generación del *Broadcast Transport Stream* se las hizo en base a 3 archivos que contienen el *Transport Stream*. Cada uno de los archivos generados con el *software Analyzer & Generator BTS* contiene diferentes parámetros de modulación, tasa de codificación, intervalo de guarda y modo, ya que solo de esta manera se verifica la versatilidad y el funcionamiento de la aplicación.

Es necesario aclarar que para generar los archivos que luego serán sometidos a pruebas de transmisión y recepción, se debe verificar que los parámetros configurados para cada capa jerárquica sean los correctos, es decir que entreguen una tasa binaria suficiente para soportar la transmisión de los paquetes asignados a dicha capa, aunque la aplicación por sí sola bloquea la opción de guardar el archivo cuando una de las capas no soporta la tasa binaria asignada a cada una.

Una de las ventajas de este software, es que la ventana del generador además de permitir generar el *Broadcast Transport Stream*, muestra en pantalla la información

de las tablas PMT del archivo que contiene el *Transport Stream*, es decir las lista de programas y los flujos elementales de cada uno con sus respectivos PID, además permite hacer la búsqueda de paquetes por su ubicación y PID.

Para cada archivo se realiza la generación de 2 *Broadcast Transport Stream*, en el primero de ellos se asignan todos los paquetes a la misma capa jerárquica, mientras tanto en el segundo se realiza las distribución de los paquetes en las 2 ó 3 capas jerárquicas.

PRUEBA I

La primera prueba se la hace en base al archivo con el nombre *vice.ts*, los parámetros configurados para la generación de los 2 *Broadcast Transport Stream* son mostrados en la Tabla 4.1.

Tabla 4.1: Parámetros de transmisión para el archivo *vice.ts*

Parámetro		BTS 1	BTS 2
Modo		3	2
Intervalo de guarda		1/8	1/32
Modulación	Capa A	64-QAM	64-QAM
	Capa B	Ninguna	QPSK
	Capa C	Ninguna	Ninguna
Tasa de codificación interna	Capa A	3/4	1/2
	Capa B	Ninguna	1/2
	Capa C	Ninguna	Ninguna
Número de segmentos	Capa A	13	12
	Capa B	0	1
	Capa C	0	0

Para la prueba de generación del BTS1 se detalla el proceso paso a paso, desde la apertura del archivo que contiene el *Transport Stream* hasta la finalización de la generación del *Broadcast Transport Stream*. La Figura 4.1, muestra el proceso de selección del archivo que será usado como base de la generación.

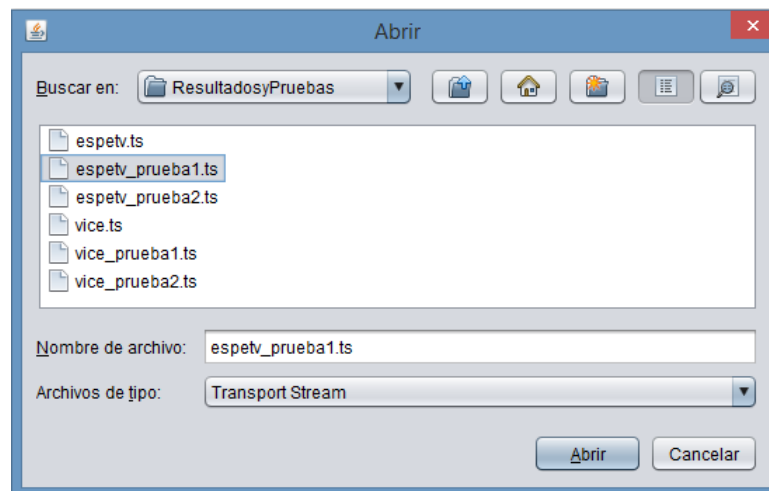


Figura 4.1: Selección del *Transport Stream*

Como se observa en la Figura 4.1, el *software* realiza el filtrado del tipo de archivos disponibles para la apertura, desplegando en pantalla solo los archivos con extensión *.ts*, es decir inhabilita la posibilidad de abrir archivos con otra extensión.

Debido a que los archivos que contienen el *Broadcast Transport Stream* también tienen la misma extensión, se ha bloqueado la apertura de este tipo de archivos desplegando en pantalla un mensaje como el que se muestra en la Figura 4.2.

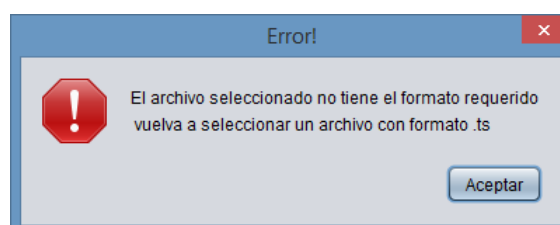


Figura 4.2: Error en la selección del archivo

Luego de haber elegido correctamente el archivo que contiene el *Transport Stream*,

se procede a configurar los parámetros mostrados en la Tabla 4.1. Hay que mencionar que el botón que permite guardar el archivo permanecerá bloqueado mientras que todos los parámetros no hayan sido correctamente configurados, desplegando en pantalla el tipo de error que se está generando. La Figura 4.3 muestra un tipo de error que puede ser generado cuando no se configuran bien estos parámetros.

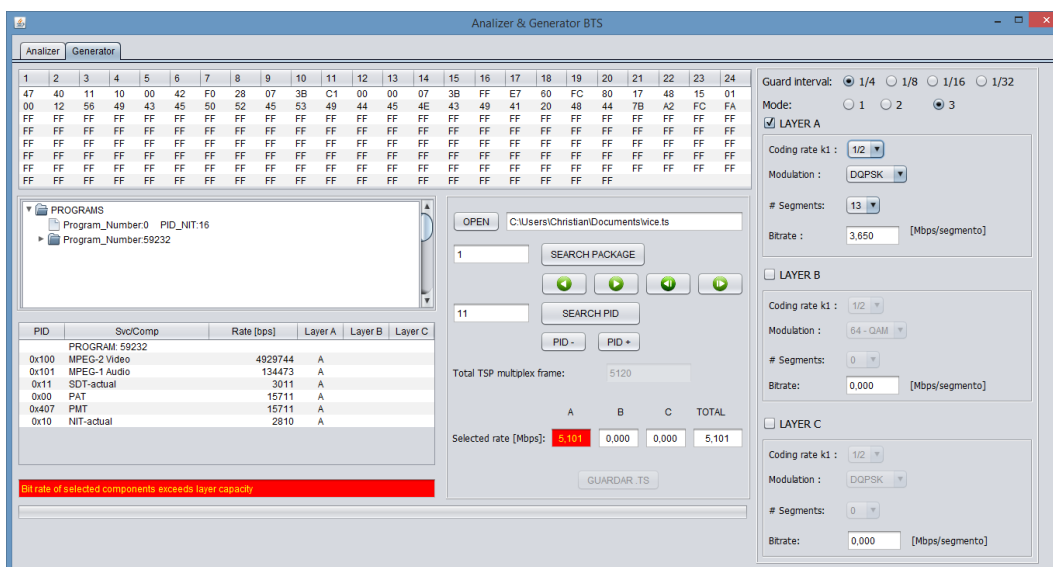


Figura 4.3: Error en los parámetros de configuración.

Como se observa en la Figura 4.3, el error mostrado en el recuadro rojo indica que la tasa de bit de los paquetes asignados a la capa A, es mayor a la tasa de bit que soporta esta capa. De la misma forma se pueden presentar otro tipo de errores, los cuales impedirán que el *Broadcast Transport Stream* pueda ser guardado.

Una vez que todos los parámetros hayan sido correctamente configurados se habilitará el botón que permite guardar el archivo, y el recuadro que aparecía en rojo, cambiará su color e indicará que todos los parámetros ISDB-T son correctos. La Figura 4.4 muestra, la correcta configuración de parámetros para la generación de BTS1.

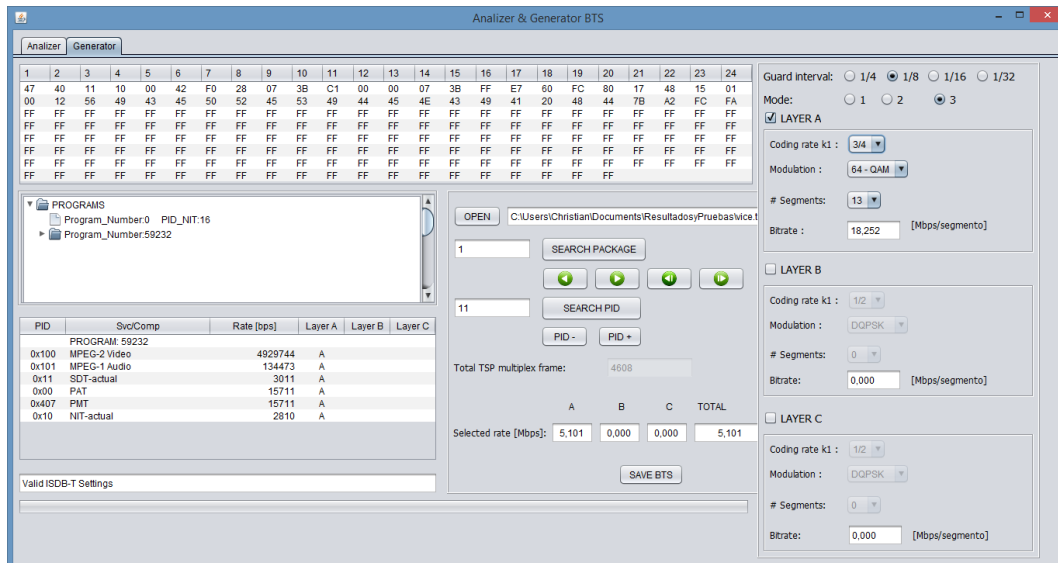


Figura 4.4: Configuración de parámetros del *Broadcast Transport Stream*.

Luego de haber seleccionado la ubicación y el nombre del archivo que contendrá el *Broadcast Transport Stream*, el software procederá a generar el archivo. El proceso de almacenamiento podrá ser verificado en la barra de progreso de la interfaz de usuario tal como se muestra en la Figura 4.5.

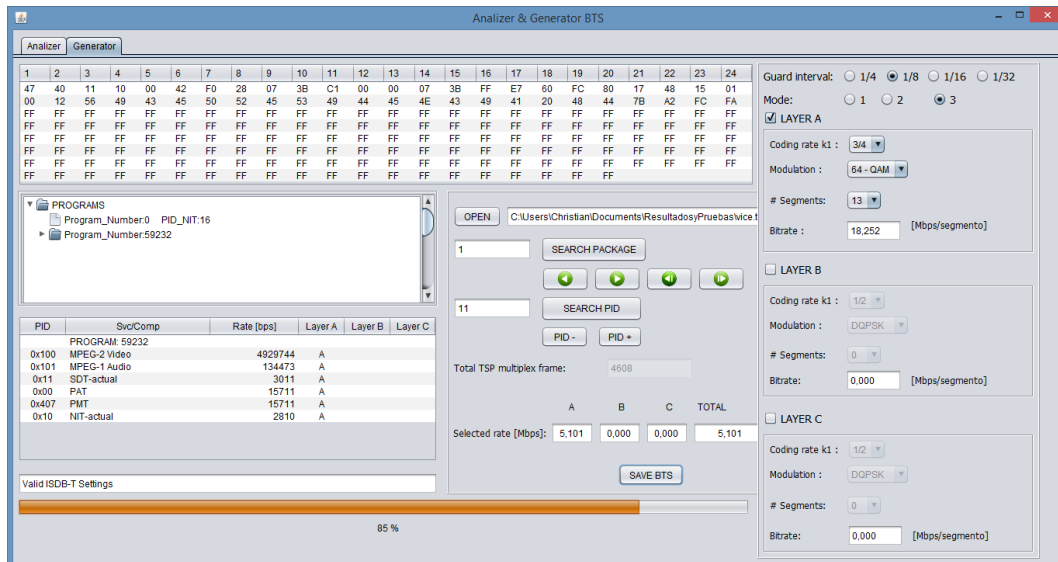


Figura 4.5: Progreso del almacenamiento del *Broadcast Transport Stream*.

Una vez finalizado el proceso y sin haber habido ningún error durante el almace-

namiento, aparecerá un mensaje en pantalla como el que se muestra en la Figura 4.6, indicando que el Broadcast Transport Stream ha sido generado exitosamente.

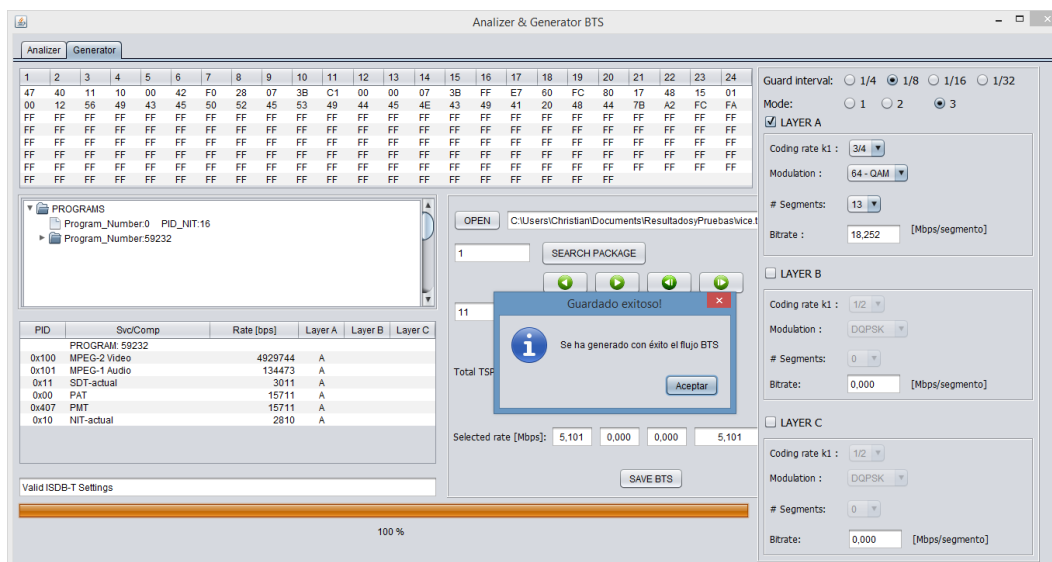


Figura 4.6: Generación del *Broadcast Transport Stream*.

Para el siguiente archivo solo se mostrará las capturas de pantalla de la configuración de los parámetros y del almacenamiento exitoso del *Broadcast Transport Stream*.

La Figura 4.7 muestra la configuración de los parámetros de transmisión mostrados en la Tabla 4.1 para el BTS2.

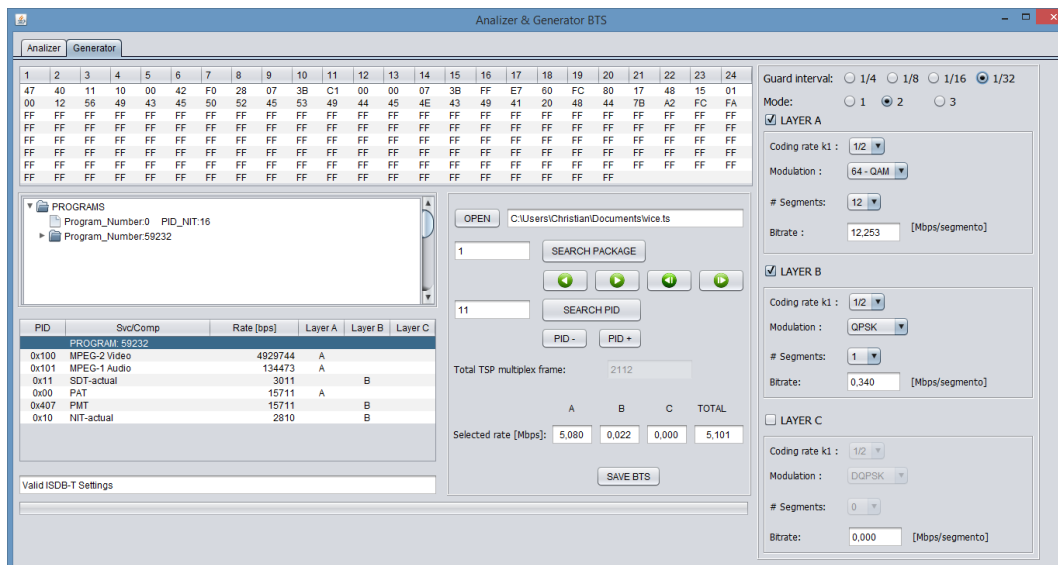


Figura 4.7: Generación del BTS2.

Igualmente que en el archivo anterior, una vez finalizado el proceso de almacenamiento se presenta en pantalla un mensaje como el mostrado en la Figura 4.9.

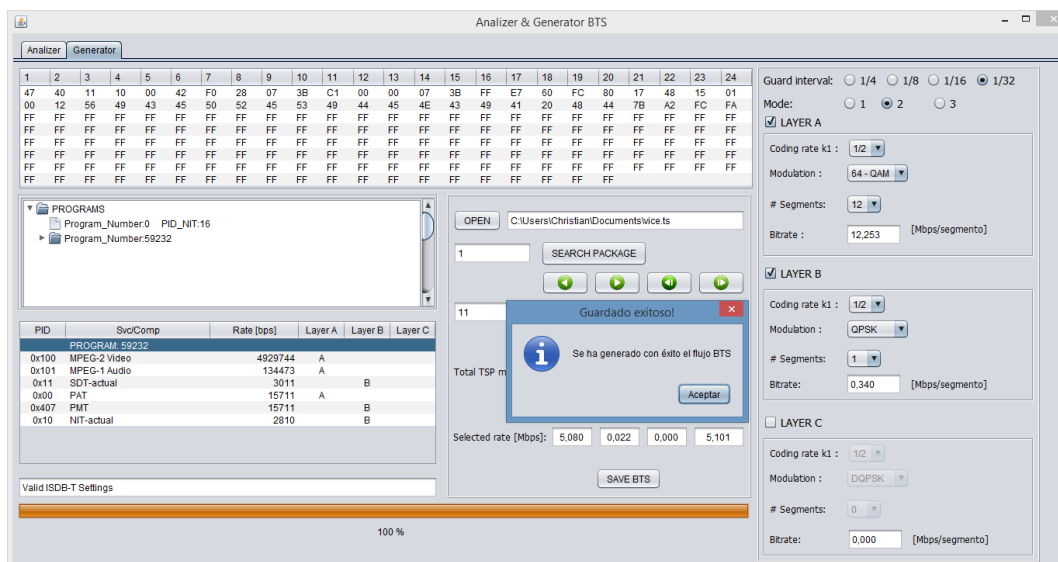


Figura 4.8: Configuración de parámetros del BTS2.

De aquí en adelante, para los archivos restantes solo se indicara cuales fueron los parámetros de transmisión escogidos, ya que el proceso de generación será el mismo

mostrado en la prueba I.

PRUEBA II

La segunda prueba se la hace en base al archivo con el nombre *espetv.ts*, los parámetros configurados para la generación de los 2 *Broadcast Transport Stream* son mostrados en la Tabla 4.2.

Tabla 4.2: Parámetros de transmisión para el archivo *espetv.ts*

Parámetro		BTS 1	BTS 2
Modo		1	2
Intervalo de guarda		1/32	1/16
Modulación	Capa A	16-QAM	16-QAM
	Capa B	Ninguna	64-QAM
	Capa C	Ninguna	16-QAM
Tasa de codificación interna	Capa A	2/3	5/6
	Capa B	Ninguna	2/3
	Capa C	Ninguna	3/4
Número de segmentos	Capa A	13	11
	Capa B	0	1
	Capa C	0	1

PRUEBA III

La tercera prueba se la hace en base al archivo con el nombre *papaya.ts*, los parámetros configurados para la generación de los 2 *Broadcast Transport Stream* son mostrados en la Tabla 4.3.

Tabla 4.3: Parámetros de transmisión para el archivo papaya.ts

Parámetro		BTS 1	BTS 2
Modo		1	2
Intervalo de guarda		1/16	1/8
Modulación	Capa A	16-QAM	64-QAM
	Capa B	Ninguna	16-QAM
	Capa C	Ninguna	QPSK
Tasa de codificación interna	Capa A	7/8	7/8
	Capa B	Ninguna	3/4
	Capa C	Ninguna	5/6
Número de segmentos	Capa A	13	10
	Capa B	0	2
	Capa C	0	1

4.2 Pruebas de análisis

Las pruebas de análisis se las hace en base a uno de los *Broadcast Transport Stream* generados anteriormente. Se analizará el campo ISDB-Info y el paquete IIP del segundo *Broadcast Transport Stream* de la Tabla 4.2, ya que este archivo muestra la distribución de sus paquetes en las 3 capas jerárquicas, verificando de esta forma si la generación ha sido exitosa.

Antes de mostrar el análisis de dicho archivo, en la Figura 4.9 se presenta la distribución de paquetes para este archivo, realizada al momento de la generación.

PID	Svc/Comp	Rate [bps]	Layer A	Layer B	Layer C
PROGRAM: 59232					
0x811	MPEG-2 Video	8271288	A		
0x81B	MPEG-1 Audio	187957	A		
0x7D1	Private Sections	3055		B	
0x7D4	13818-6 DSM CC	2000011	A		
0x00	PAT	15098			C
0x407	PMT	15098			C
0x11	SDT-actual	3055			C
0x10	NIT-actual	3055			C

Figura 4.9: Asignación de paquetes para BTS2.

Las Figuras 4.10, 4.11 y 4.12 muestran la asignación de los paquetes del *Broadcast Transport Stream*, a cada una de las capas jerárquicas.

The screenshot shows the 'Analyzer & Generator BTS' software. The main window is divided into several sections. At the top, there is a hex dump of a packet. Below that, there is a search section with a 'PID' field containing '7D4' and a 'SEARCH PACK...' button. The 'ISDB_Info' section displays various parameters for the selected packet, including 'TMCC Identifier: DIGITAL TV', 'Buffer reset control flag: NORMAL STATE', 'Control flag for emergency broadcasting: NORMAL STATE', 'Initialization timing head packet flag: NORMAL', 'Frame head packet flag: NORMAL TSP', 'Frame indicator: SYNC SO: 0011010111101110', 'Layer indicator: TSP LAYER A', 'Count down index: WITHOUT CHANGES OF PARAMETERS', 'AC data invalid flag: NO DATA AC', 'AC data effective bytes: NO DATA AC', and 'TSP Counter: 3'. The 'Layer indicator' field is highlighted with a red box and labeled 'CAPA'. The 'PID' field in the search section is also highlighted with a red box and labeled 'PID'. The right sidebar shows a tree view of the program structure, including 'Program_Number0 PID:NIT', 'Program_Number59232', 'Stream Type 1: MPEG-2 high', 'Stream Type 2: MPEG-1 audio', 'Stream Type 3: MPEG-2 Private Sections', and 'Stream Type 4: ISM/EC 13818-6 DSM-CC'.

Figura 4.10: Asignación de los paquetes a la capa A.

Como se observa en las Figuras 4.10, 4.11 y 4.12, la asignación de los paquetes se lo hace de acuerdo a su PID según lo descrito en la Figura 4.9. Además de esto, la ventana del analizador permite visualizar toda la información del campo ISDB-Info, la distribución de programas dentro del *Broadcast Transport Stream*, y también brinda la opción de realizar la búsqueda de paquetes de acuerdo a su PID y a su ubicación dentro del archivo.

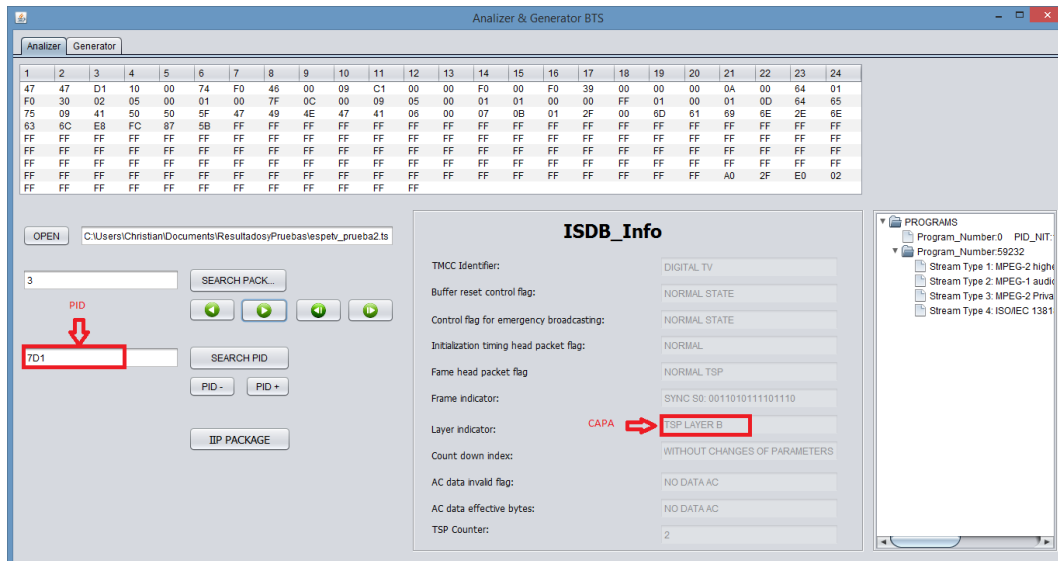


Figura 4.11: Asignación de los paquetes a la capa B.

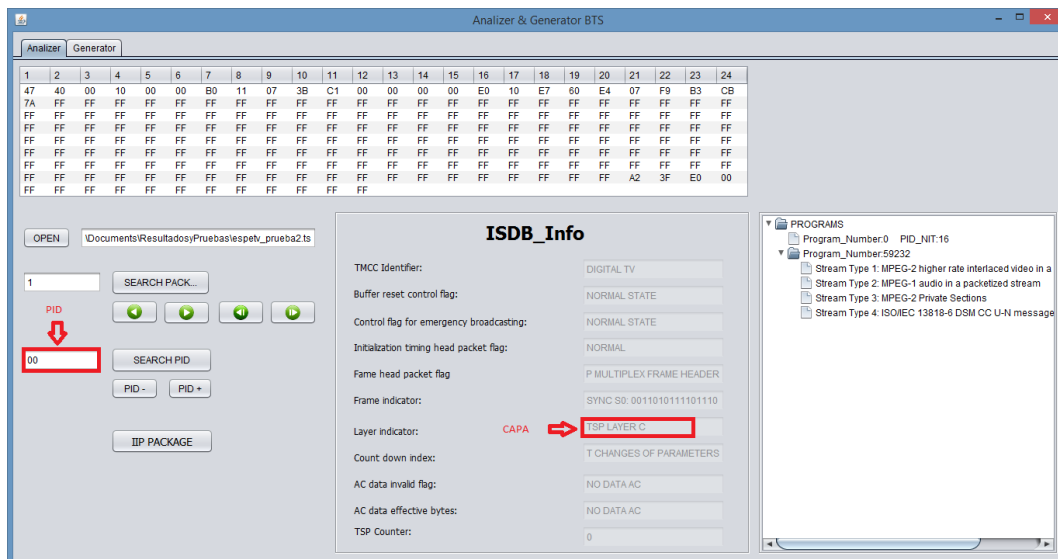


Figura 4.12: Asignación de los paquetes a la capa C.

Al dar clic sobre el botón *IIP PACKAGE*, se despliega la información del paquete IIP. La Figura 4.13 muestra la interfaz de usuario que tendrá esta ventana.

Contrastando la Figura 4.13 con la tabla 4.2 se verifica que los parámetros de transmisión coinciden exactamente para el BTS2, es decir se comprueba que el proceso de generación del *Broadcast Transport Stream* para este archivo ha sido correcta.

IIP PACKAGE

TMCC synchronization word:

AC data effective position:

Initialization timing indicator:

Mode:

Guard interval:

TMCC Information

System ID: Count down index:

Alert broadcasting:

Partial reception flag:

	Layer A	Layer B	Layer C
Modulation:	<input type="text" value="16 QAM"/>	<input type="text" value="64 QAM"/>	<input type="text" value="16 QAM"/>
Coding rate:	<input type="text" value="5/6"/>	<input type="text" value="2/3"/>	<input type="text" value="3/4"/>
Time interleaving:	<input type="text" value="16"/>	<input type="text" value="16"/>	<input type="text" value="16"/>
Segments number:	<input type="text" value="11 segments"/>	<input type="text" value="1 segments"/>	<input type="text" value="1 segments"/>

Figura 4.13: Paquete IIP.

4.3 Pruebas de transmisión

El escenario usado para la transmisión de los archivos generados con el *software Analyzer & Generator BTS* se presenta en la Figura 4.14.

El escenario mostrado en la Figura 4.14, se realizó con equipos de Laboratorio de Televisión Digital de la Universidad de las Fuerzas Armadas - ESPE. Para la transmisión de los archivos generados fue necesario el *software Stream Xpress* el cual con-

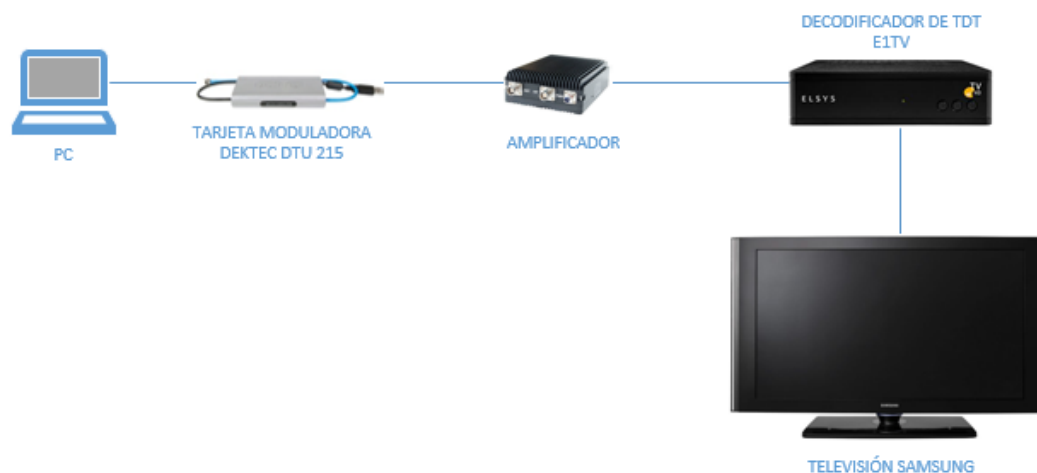


Figura 4.14: Escenario de pruebas para la transmisión y recepción

trola a la tarjeta moduladora DEKTEC DTU 215 y además muestra en su interfaz información de los campos del *Broadcast Transport Stream* permitiendo así realizar un nuevo análisis de los BTS generados.

La Figura 4.15 muestra la interfaz gráfica de usuario del *software Stream Xpress*, además se observa que el archivo que contiene el *Broadcast Transport Stream* analizado en la Sección 4.2, se ha abierto para ser transmitido.

Comparando las Figuras 4.13 y 4.15, que contienen los parámetros del mismo BTS, se puede observar que dichos parámetros coinciden en las dos ventanas, pudiendo de esta manera afirmar que el *software* desarrollado es totalmente funcional. En la Figura 4.15 también se observa que se ha activado la casilla de TMCC info, la cual permitirá usar la información configurada por el *software Analyzer & Generator BTS* al momento de la generación del archivo.

Una vez abierto el archivo, activada la casilla de TMCC info y seleccionado el canal en el cual queremos que se transmita la señal, se da clic sobre el botón *play* de la ventana y se procede a sintonizar el canal en el receptor. La Figura 4.16 muestra la recepción de la señal transmitida en el canal seleccionado.

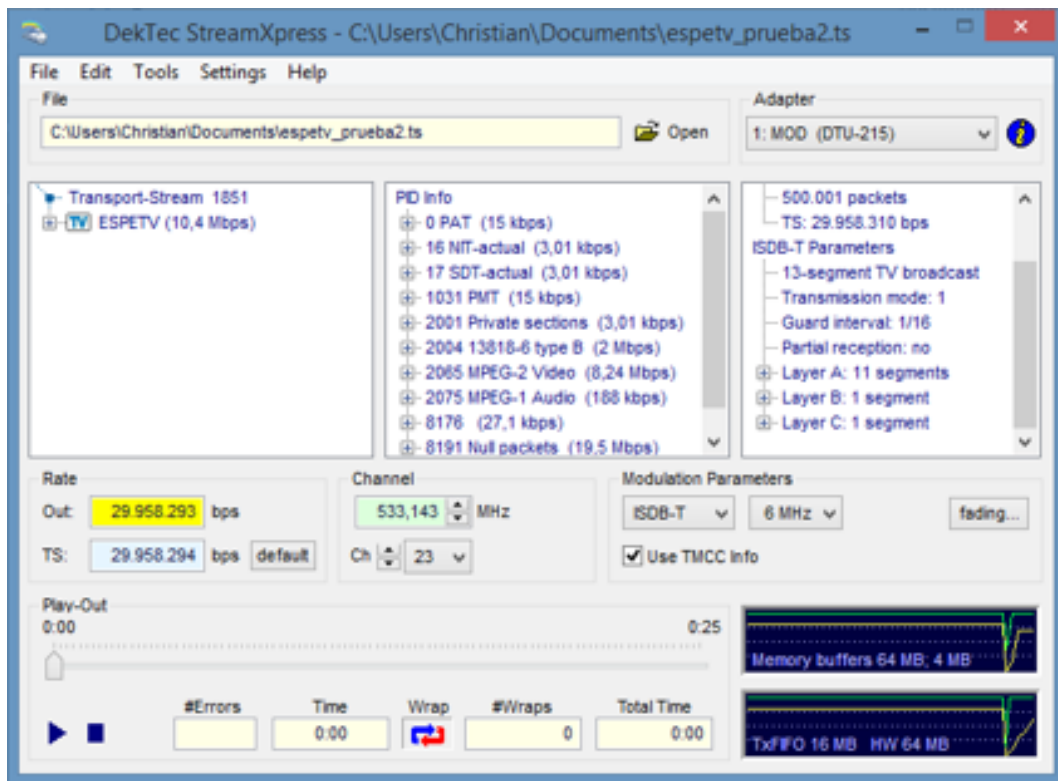


Figura 4.15: Interfaz gráfica de *Stream Xpress*

Para validar que la señal transmitida tenga el ancho de banda de 6 MHz establecido en el estándar ISDB-Tb se envía la señal de salida del amplificador al analizador de televisión digital Agilent CXA.

La Figura 4.17 muestra que efectivamente la señal que está siendo transmitida está ocupando un ancho de banda de 6 MHz, además se observa que la frecuencia de transmisión 575 MHz que es aproximadamente igual a la frecuencia teórica de transmisión igual a 575,143 MHz.



Figura 4.16: Recepción de la señal del *Broadcast Transport Stream*



Figura 4.17: Verificación de ancho de banda

CAPÍTULO 5

Conclusiones y Recomendaciones

5.1 Conclusiones

- Se desarrolló un *software* de generación del Flujo Transporte para *Broadcast* de televisión digital terrestre bajo el lenguaje de programación Java, permitiendo además realizar el análisis de dicho flujo.
- Se obtuvo la tasa binaria de cada una de las tablas y de los flujos elementales presentes en el *Transport Stream*, pudiendo de esta forma controlar la asignación de los paquetes a cada una de las capas jerárquicas.
- Se obtuvo la lista de programas con su respectivo PID y flujos de audio, video y datos que conforman cada uno.
- Se generó exitosamente el paquete IIP y la información del Campo ISDB-Info de cada paquetes TSP.
- Se analizó el campo ISDB-Info y el paquete IIP de cada uno de los *Broadcast Transport Stream* generados, y se verificó que contengan la información configurada por el usuario al momento de su generación.
- En base a la investigación y análisis de la tablas de estándar, se utilizó la información de la PAT, PMT y NIT para el desarrollo del software.

- Se realizó pruebas de transmisión de los *Broadcast Transport Stream* generados bajo diferentes parámetros de modulación, intervalo de guarda, modo, relación de codificación interna y esquemas de asignación de las capas jerárquicas, y se verificó que cada una de las transmisiones sea exitosa.
- El análisis de todos los *Broadcast Transport Stream* generados, se lo realizó en el software desarrollado y en el *software Stream Xpress*, obteniendo los mismos resultados en los dos.
- Se realizó una interfaz gráfica de usuario, tratando de que el *software* sea lo más dinámico y amigable, y no cause ningún tipo de confusión a los usuarios.

5.2 Recomendaciones

- Se recomienda que la tasa binaria de cada capa jerárquica sea notablemente superior a la tasa binaria de los paquetes asignados a cada capa, ya que cuando no sucede esto la transmisión del *Broadcast Transport Stream* tendrá pausas. Hay que aclarar que estas pausas o cortes en el video también se presentan cuando se transmite el archivo original del *Transport Stream*, configurando los parámetros de transmisión en el *software Stream Xpress*.
- Se recomienda no pausar o detener el proceso de generación de *Broadcast Transport Stream*, mientras el *software* no haya indicado que el proceso ha terminado, ya que si se detiene el proceso el archivo generado no estará completo y no podrá ser transmitido.
- El software desarrollado permite realizar la generación del Flujo de Transporte para *Broadcast* de archivos con una tasa binaria igual a 29.96 Mbps que es la tasa recomendada para la transmisión. En caso de tener archivos con una tasa inferior a la mencionada el *Broadcast Transport Stream* generado tendrá una pér-

dida de sincronismo al momento de su transmisión, debido a que dicho software no contempla la inserción de paquetes nulos en el archivo generado.

- Las tasas binarias de cada uno de los flujos del *Transport Stream* usados para la generación, tardarán un momento en aparecer al momento de abrir un archivo, éste se debe a que el *software* debe realizar clasificar los PID de todos los paquetes que conforman el *Transport Stream*. Mientras no aparezcan las tasas de cada flujo, no se podrá realizar la generación del *Broadcast Transport Stream*.
- Se recomienda desactivar las capas jerárquicas en las cuales no se ha asignado ningún paquete.

5.3 Trabajos Futuros

El desarrollo de este proyecto deja como principal propuesta el desarrollo de un nuevo *software* que permita la generación del *Broadcast Transport Stream* de archivos con tasas binarias diferentes de 29.96 Mbps. El software *Analizer & Generator BTS* es fácilmente adaptable a la incorporación de nuevos métodos y clases ya que en él se implementa el principio de Modularidad planteado en el capítulo 3, es por eso que no se descarta la posibilidad de trabajar sobre este código para realizar lo que se deja planteado. El principal objetivo es realizar un software que se adapte a cualquier ambiente de transmisión y permita realizar la transmisión de cualquier tipo de contenido.

BIBLIOGRAFÍA

- [1] C. Sánchez, “Propuesta para la implementación de un sistema de televisión digital terrestre para pequeños radiodifusores con el uso de soluciones en software.”
- [2] M. de Telecomunicaciones y Sociedad de la Información. Televisión digital terrestre en el Ecuador. [Online]. Available: <http://www.telecomunicaciones.gob.ec/television-digital-terrestre-en-el-ecuador/>
- [3] J. A. Parreño Escobar and S. R. Ponce Mantilla, “Creación de nuevos servicios de televisión digital terrestre bajo el estándar isdb-tb para la plataforma village-flow mediante el análisis de su estructura,” Ph.D. dissertation, Universidad de las Fuerzas Armadas ESPE. Carrera de Ingeniería en Electrónica y Telecomunicaciones, 2014.
- [4] R. Sotelo, D. Durán, and J. Joskowicz, “Sistema de transmisión isdb-t,” 2011.
- [5] C. de Investigación para la Sociedad de la Información IMAGINAR. La televisión digital. [Online]. Available: http://www.imaginar.org/iicd/fichas/10_TVdigital.pdf
- [6] P. F. Sotomayor Jácome, “Análisis de los estándares de televisión digital terrestre (tdt) y pruebas de campo utilizando los equipos de comprobación técnica de la superintendencia de telecomunicaciones,” 2009.
- [7] F. Sandoval. Isdb-t e isdb-tb. [Online]. Available: http://www.slideshare.net/blog_fralbe/7-isdb

- [8] M. de Telecomunicaciones del Ecuador. Proceso de implementación de la televisión digital en el Ecuador. [Online]. Available: http://www.telecomunicaciones.gob.ec/wp-content/uploads/downloads/2015/02/PRESENTACION%CC%81N_TDT_MINTEL-Febrero-2015.pdf
- [9] M. S. Alencar, “Digital television systems,” 2009.
- [10] R. V. Haro Báez, “Evaluación del desempeño y optimización del sistema de televisión digital terrestre isdb-tb e iptv de la espe,” Ph.D. dissertation, SANGOLQUÍ/ESPE/2012, 2012.
- [11] N. O. Pisciotta, “Transmisión de televisión digital terrestre en la norma isdb-tb,” 2013.
- [12] A. D. Gutiérrez, “Flujos de programa y de transporte mpeg-2 aplicación a dvb,” 2001.
- [13] ABNT, “Televisión digital terrestre multiplexación y servicios de información (si) parte 1: Si del sistema de radiodifusión,” 2007.
- [14] —, “Televisión digital terrestre multiplexación y servicios de información (si) parte 2: Estructura de datos y definiciones de la información básica de si,” 2009.
- [15] N. O. Pisciotta, “Remultiplexor isdb-tb,” pp. 57–68, 2014.
- [16] ABNT, “Abnt nbr 15608-1, guía de operación, parte 1: Sistema de transmisión - guía para implementación de abnt nbr 15601: 2007,” 2008.
- [17] J. J. M. Navarro. Arquitecturas software. [Online]. Available: http://babel.ls.fi.upm.es/~fred/sbc/arquitecturas_sw.pdf
- [18] N. M. T. Villanueva. Manejo de archivos. [Online]. Available: <http://www.slideshare.net/chicaGeekGPLMX/manejo-de-archivos-en-java?related=1>

- [19] F. de Ingeniería en Electricidad y Computación. Archivos de acceso aleatorio. [Online]. Available: <http://www.slideshare.net/jent46/archivos-aleatorios-en-java-presentation>
- [20] E. B. Pantoja, “El patrón de diseño modelo-vista-controlador (mvc) y su implementación en java swing,” p. 493, 2004.

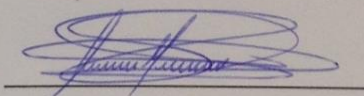
UNIVERSIDAD DE LAS FUERZAS ARMADAS - ESPE
INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

ACTA DE ENTREGA

El presente proyecto fue entregado en el Departamento de Eléctrica y Electrónica, y reposa en los archivos desde:

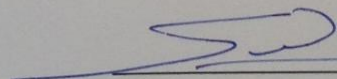
Sangolquí, 5 de Agosto de 2015.

Elaborado por:



Christian Ricardo Mancheno Arcos

Autoridad:



Ing. Paúl Bernal Oñate MSc.

DIRECTOR DE LA CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES (E).

