DEPARTAMENTO DE ELÉTRICA Y ELECTRÓNICA

CARRERA DE INGENIERÍA EN ELECTRÓNICA Y TELECOMUNICACIONES

TRABAJO DE TITULACIÓN PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELETRÓNICA Y TELECOMUNICACIONES

TEMA: ESTUDIO, ANÁLISIS E IMPLEMENTACIÓN DE UN SOFTWARE PARA CONSTRUIR UN EXTRACTOR Y CONSTRUCTOR DE CARRUSEL DE DATOS INCLUIDO EN EL FLUJO DE TRANSPORTE MPEG-2 TS DE TELEVISIÓN DIGITAL

AUTOR: NÚÑEZ NARVÁEZ ÉDISON ANDRÉS

DIRECTOR: DR. OLMEDO CIFUENTES, GONZALO

SANGOLQUÍ

2016

CERTIFICACIÓN

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

CERTIFICACIÓN

Certifico que el trabajo de titulación, **"ESTUDIO, ANÁLISIS E IMPLEMENTACIÓN DE UN SOFTWARE PARA CONSTRUIR UN EXTRACTOR Y CONSTRUCTOR DE CARRUSEL DE DATOS INCLUIDO EN EL FLUJO DE TRANSPORTE MPEG-2 TS DE TELEVISIÓN DIGITAL"** realizado por el señor **EDISON ANDRÉS NÚÑEZ NARVÁEZ**, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar al señor **EDISON ANDRÉS NÚÑEZ NARVÁEZ** para que lo sustente públicamente.

Sangolquí, 24 de Agosto del 2016

Dr. Gonzalo Olmedo

DIRECTOR

AUTORÍA DE RESPONSABILIDAD

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

AUTORÍA DE RESPONSABILIDAD

Yo, **EDISON ANDRÉS NÚÑEZ NARVÁEZ**, con cédula de identidad N° 1715824262 declaro que este trabajo de titulación **"ESTUDIO, ANÁLISIS E IMPLEMENTACIÓN DE UN SOFTWARE PARA CONSTRUIR UN EXTRACTOR Y CONSTRUCTOR DE CARRUSEL DE DATOS INCLUIDO EN EL FLUJO DE TRANSPORTE MPEG-2 TS DE TELEVISIÓN DIGITAL"** ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

Consecuentemente declaro que este trabajo es de mi autoría, en virtud de ello me declaro responsable del contenido, veracidad y alcance de la investigación mencionada.

Sangolquí, 24 de Agosto del 2016

EDISON ANDRÉS NÚÑEZ NARVÁEZ

C.C.: 1721237475

AUTORIZACIÓN

ESPE

UNIVERSIDAD DE LAS FUERZAS ARMADAS

INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE ELÉCTRICA Y ELECTRÓNICA

CARRERA DE ELECTRÓNICA Y TELECOMUNICACIONES

AUTORIZACIÓN

Yo, **EDISON ANDRÉS NÚÑEZ NARVÁEZ**, autorizo a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación **"ESTUDIO, ANÁLISIS E IMPLEMENTACIÓN DE UN SOFTWARE PARA CONSTRUIR UN EXTRACTOR Y CONSTRUCTOR DE CARRUSEL DE DATOS INCLUIDO EN EL FLUJO DE TRANSPORTE MPEG-2 TS DE TELEVISIÓN DIGITAL"** cuyo contenido, ideas y criterios son de mi autoría y responsabilidad.

Sangolquí, 24 de Agosto del 2016

EDISON ANDRÉS NÚÑEZ NARVÁEZ

C.C.: 1721237475

DEDICATORIA

Especialmente a mi padre Édison que siempre ha sido mi ejemplo a seguir a pesar de haberse ido de este mundo muy temprano y a mi madre Teresa por confiar tanto en mí y que me ha sabido guiar toda la vida, más aún en los momentos difíciles. También a mis hermanas Steffy y Patricia, mis sobrinos Dominic, Leo y Hannah porque siempre han sido parte importante en mi vida.

Édison Andrés Núñez Narváez

AGRADECIMIENTO

Agradezco principalmente a mis padres Édison y Teresa que me supieron enseñar ante todo como ser una buena persona, respetuoso y consiente de la importancia de la educación académica para lograr el cumplimiento de mis metas y proyecto de vida.

A mis hermanas Steffi y Patricia, mis sobrinos Dominic, Leo y Hannah, mi mejor amigo Jonathan y gran parte de mi familia los cuales supieron entender mi sacrificio y apoyar mis decisiones, durante el duro pero gratificante camino que fue mi carrera universitaria.

A mi Director de tesis el Dr. Gonzalo Olmedo que confió en mis capacidades para realizar este proyecto y además supo guiarme de la mejor manera para lograrlo.

También agradezco mucho a mis amigos dentro y fuera de la universidad, en especial a aquellos que no solo festejaron cada uno de mis logros sino que además me ayudaron a levantar de mis tropiezos y caídas.

Édison Andrés Núñez Narváez

ÍNDICE DE CONTENIDO

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

Desde sus inicios la Televisión Digital Terrestre (TDT) trajo consigo grandes beneficios, entre los cuales se tuvo la posibilidad de transmitir información multimedia a través de red, lo que se conoce como interactividad. La TDT transmite aplicaciones interactivas a través de encapsulación de los datos en paquetes y multiplexándolos dentro del *Transport Stream* (TS) junto con otros paquetes correspondientes a otros servicios que incluye video, audio, etc. Sin embargo, para que el receptor pueda comprender y decodificar los datos, el transmisor debe convertir las aplicaciones y su contenido en un carrusel de datos y objetos, estructura mediante la cual es posible transmitirlo de forma cíclica y ordenada, facilitando su extracción, organización y ejecución en el lado del receptor. El presente trabajo muestra el diseño, implementación y la evaluación de un algoritmo que permite la extracción de la sección de datos del TS y su organización en un sistema jerárquico de archivos y directorios, en base a la norma MPEG-2 ISO-13818-6: Extensiones para el Comando y Control de Almacenamiento de Medios Digital (DSM-CC del inglés Digital Storage Media Command and Control), con el objetivo de recuperar totalmente el contenido multimedia de aplicaciones interactivas que contiene el flujo MPEG-2 TS, sin la necesidad utilizar un decodificador especializado que realice dicha tarea. Para comprobar el algoritmo se implementó un software en MATLAB, que evalúa la extracción de aplicaciones de diferentes TS. Se recuperaron satisfactoriamente todos los directorios y archivos de las aplicaciones interactivas.

PALABRAS CLAVE:

- **APLICACIONES INTERACTIVAS**
- **MPEG-2 DSM-CC**
- **CARRUSEL DE DATOS**
- **CARRUSEL DE OBJETOS**
- **TELEVISIÓN DIGITAL TERRESTRE**

ABSTRACT

From its beginning Digital Terrestrial Television (DTT) brought great benefits, including the possibility of transmitting multimedia information over network which is known as interactivity. DTT broadcast interactive applications through encapsulation of data in packets and multiplexing them within the Transport Stream (TS) along with other packets that correspond to other services including video, audio, etc. However, so that the receiver can understand and decode the data, the transmitter must convert applications and content on a carousel of data and objects, structure through which it is possible to cyclically and orderly transmit it, facilitating its removal, organization and execution on the receiver side. This paper presents the design, implementation and evaluation of an algorithm that allows the extraction of the data section of the TS and its organization in a hierarchical file system and directories, based on the MPEG-2 standard ISO-13818-6: Extensions for Digital Storage Media Command and Control (DSM-CC), in order to fully recover the media of interactive applications contained in the MPEG-2 TS stream, without the need of using a specialized decoder to perform this task. To test the algorithm, a MATLAB software was implemented, which evaluates the applications extraction of different TS. All directories and files of interactive applications were successfully recovered.

KEYWORDS:

- **INTERACTIVE APPLICATIONS**
- **MPEG-2 DSM-CC**
- **DATA CAROUSEL**
- **OBJECT CAROUSEL**
- **TERRESTRIAL DIGITAL TELEVISION**

CAPÍTULO 1

1.1. Antecedentes

La Televisión Digital Terrestre (TDT) desde sus inicios trajo consigo grandes beneficios debido a las características que esta tecnología posee, entre las cuales se tuvo la posibilidad de tener televisión móvil (mTV del inglés *mobile Television*), manejo señales de emergencia (EWBS del inglés *Emergency Warning Broadcast System*), capacidad para transmitir múltiples programas sobre el mismo ancho de banda que maneja la televisión analógica convencional, variedad en calidad de imagen (LDTV de inglés *Low Definition Television*, SDTV de inglés *Standard Definition Television* y HDTV de inglés *High Definition Television*) y transmisión de información multimedia de forma bidireccional, conocida como interactividad, causando que a nivel mundial y particularmente en países sudamericanos como Brasil tome gran importancia, ya que esto se ha visto no solo como un cambio de tecnología de análoga a digital, sino también como una forma de mejorar la calidad de vida de la población, incluso generando propuestas de políticas públicas para poder implementar y controlar de manera eficaz la televisión digital en dicho país (Zuffo, 2008).

En el caso de Ecuador, desde 26 de marzo de 2010, fecha en que se adoptó oficialmente el estándar japonés-brasileño (ISDB-Tb) para Televisión Digital Terrestre, entes gubernamentales e instituciones relacionados con el sector de las telecomunicaciones (MINTEL, SENPLADES, SENESCYT, SENATEL, SUPERTEL ahora ARCOTEL, OPERADORES) formaron el Comité Interinstitucional Técnico de Implementación de la Televisión Digital Terrestre (CITDT) para trabajar en conjunto en el proceso transitorio de implementación de este sistema esperando así garantizar el derecho a la comunicación, inclusión y equidad social de la población en general, además de poder generar fácilmente contenido de carácter educativo, salud y cultura, mejorando así la calidad de programación actual, mediante múltiples programas y aplicaciones. Telegobierno, Telesalud y Teleducación (Ministerio de Telecomunicaciones y Sociedad de la Información, 2010).

Por el momento en el Ecuador se mantiene emitiendo señales de prueba de televisión digital en la ciudades de Quito, Guayaquil, Cuenca, Ambato, Latacunga, Santo Domingo, Manta y Portoviejo hasta poder completar la fase 1 del apagón digital previsto el 31 de diciembre del 2016 (Ministerio de Telecomunicaciones y Sociedad de la Información, 2010). Aun así en cuestión de interactividad son pocas las pruebas que se han realizado principalmente por la falta de infraestructura que poseen los canales de televisión para transmitir interactividad.

El mecanismo utilizado para transportar el contenido multimedia de la aplicaciones interactivas a través del flujo de transporte (TS del inglés *Transport Stream*) MPEG-2, es mediante la encapsulación de la información dentro de un carrusel de datos el cual mantiene una estructura propia del Comando y Control de Almacenamiento de Medios Digital (DSM-CC del inglés *Digital Storage Media Command and Control*), que no es más que un conjunto de herramientas y protocolos que proveen una manera de encapsular datos e información para que posteriormente al ser transmitidos en una red de tipo usuario a usuario (U-U del inglés *User to User*) o usuario a red (U-N del inglés *User to Network*), estos puedan ser explorados, seleccionados, descargados, controlados y organizados en la forma jerárquica de archivos y directorios que las aplicaciones requieran (ISO/IEC, 1998).

Aunque inicialmente el DSM-CC fue creado con el propósito de manejar transmisión de video vía satélite o terrestre con contenido interactivo e IPTV, debido a su capacidad de proporcionar un amplio rango de funcionalidad para apoyar las tecnologías multimedia, hoy en día también se puede tomar como una forma de transmisión de datos para una variedad de servicios o aplicaciones, independientemente del tipo o topología de red que estos manejen (ISO/IEC, 1998). Dando al cliente (en el caso del sistema TDT) la posibilidad de simular una conexión con el servidor, de tal forma que este pueda solicitar la descarga de un fichero en particular, aun cuando la conexión por naturaleza de un sistema *broadcast* sea exclusivamente unidireccional de servidor a cliente, tomando en cuenta que el transmisor enviará repetidamente y en forma cíclica los datos, durante el tiempo que dure la transmisión del TS (Morris, 2011).

La utilización del DSM-CC está definida en dentro de la norma ISO-13818-6: Extensiones para el DSM-CC, la cual provee una visión de los protocolos que usa el DSM-CC, sobre una red que maneje canales con multiplexación de paquetes de tipo MPEG 2 TS o de forma independiente, dependiendo de la aplicación o servicio (Crinon, 1997).

1.2. Justificación e Importancia

Dentro del marco regulatorio de las telecomunicaciones en el Ecuador se encuentra la Ley de Radiodifusión y Televisión que en su Art. 5-E establece que se debe regular y controlar, en todo el territorio nacional, la calidad artística, cultural y moral de los actos o programas de las estaciones de radiodifusión y televisión (ARCOTEL, 1995). Así mismo la normativa de la ley mencionada en su Art. 48 brinda ciertas pautas en cuanto a la programación indicando que en el horario que va desde las 06h00 hasta las 21h00, se evitará mostrar escenas o imágenes de violencia, crueldad, actos sexuales explícitos o de promiscuidad con el objetivo de prevenir vicios u otras desviaciones de la conducta individual o social. También evitar emitir comerciales con imágenes de cigarrillos o bebidas alcohólicas que de alguna manera estén dirigidas directa o indirectamente a menores de edad (ARCOTEL, 1996).

Cabe mencionar que las escenas o imágenes dentro del contexto de lo que es el Sistema de Televisión Digital Terrestre, no solo se debe tomar en cuenta las que muestra audio o video sino también las sección DSM-CC, la misma que trae consigo información multimedia y ficheros los cuales también deberían ser controlados y regulados mediante los artículos antes mencionados.

El organismo encargado de hacer cumplir la Ley de Radiodifusión y Televisión, es la Agencia de Regulación y Control de las Telecomunicaciones (ARCOTEL), el cual hoy en día posee tecnología y mecanismos para monitorear y controlar las 24 horas las diferentes emisiones de programas de radio y televisión análoga o digital. Entre estos mecanismos se encuentran estaciones móviles las cuales cuentan con

matrices de diferentes antenas para poder monitorear todos los servicios de telecomunicaciones incluyendo el de televisión digital.

Sin embargo en lo que respecta a televisión digital los equipos de monitoreo de la ARCOTEL, solo permiten observar las tablas de información específica de programa (PSI del inglés *Program Specific Information*) o información de servicio (SI del inglés *Service Information*) del TS y desde luego se puede observar también si la señal recibida cuenta o no cuenta con interactividad, mas no poseen una manera de descargar y reconstruir los archivos, imágenes o datos correspondientes a la interactividad, quedando así incompleta la labor de controlar y regular esta parte de la TDT, incluso contando con un decodificador específico (Set-top box) que ejecute las aplicaciones interactivas, los organismo de supervisión únicamente se limitarían a observar su funcionamiento.

Como se mencionó anteriormente en cuestión de regulación y control de contenido de las aplicaciones interactivas, la capacidad que se tenga de poder visualizar y manipular dicha información es muy importante, sin embargo no solo en esto se limita su aplicación e importancia, ya que el poder extraer y visualizar los datos que transporta el TS de TDT, permite también realizar un análisis más profundo de ellos, con el objetivo de mejorar la calidad de programación del código fuente de la aplicación, o que el formato o tipo del contenido multimedia sea el adecuado para un mejor desempeño de la aplicaciones al momento de su ejecución.

Entonces la importancia de generar un *software* que sea capaz de manipular la sección DSM-CC y extraer los datos, radica en no limitarse solo a observar y verificar el funcionamiento de la aplicación, sino también en poder visualizar de forma ordenada todo el contenido de la misma (imágenes, archivos de texto, código fuente de la aplicación y otro ficheros que podrían estar dentro del DSM-CC) dentro de un sistema jerárquico de archivos y directorios, facilitando el proceso de regulación, control o análisis de la información referente a la transmisión de datos por TDT.

1.3. Alcance

El presente trabajo comprende el desarrollo de un algoritmo y su implementación en un *software* con interfaz gráfica de usuario programando en MATLAB, el cual mediante las especificaciones del estándar MPEG 2 DSM-CC, permita descargar y discriminar los datos de la aplicación interactiva presentes en un TS. Cabe destacar que se usó exclusivamente el modelo de carrusel de datos de una capa en el algoritmo, concepto que es explicado y justificado su uso en el Capítulo 2.

El *software* desarrollado permite escoger el TS del cual se desea extraer las secciones DSMCC. Este a través del análisis de las tablas PSI, determina qué valor es el que corresponde al identificador de los paquetes (PID del inglés *Packet Identification*) correspondientes que transportan los datos de las aplicaciones interactivas. Posteriormente el programa utilizará únicamente los paquetes de TS necesarios para poder reconstruir el directorio con sus respectivos archivos. Además de la descarga y reconstrucción de datos, al finalizar la operación, el *software* genera un reporte detallado y resumido el cual contiene información referente al proceso de extracción como por ejemplo: número de bloques de descarga y objetos encontrados, el tipo de objeto y su tamaño en bytes, clave de objeto, etc.

1.4. Objetivos

1.4.1. General

- Diseñar e implementar un *software*, programado en MATLAB que permita descargar, reconstruir y organizar los datos de información multimedia de la sección DSM-CC, contenida dentro del *Transport Stream* (TS).

1.4.2. Específicos

- Estudiar y analizar el estado el arte respecto a las formas existentes de manipular el DSM-CC.
- Estudiar y analizar la función que cumple el carrusel de datos y el carrusel de objetos dentro de un proceso de transporte de datos en un sistema TDT.
- Diseñar el algoritmo de discriminación y extracción de paquetes del DSM-CC multiplexados dentro del TS, mediante el análisis de la tabla de mapa de programa

(PMT del inglés *Program Map Table*), en la cual se puede identificar el PID de dichos paquetes.

- Estudiar y analizar la estructura de un carrusel de datos de una capa y establecer diferencias con respecto al carrusel de datos de dos capas.
- Diseñar el algoritmo de descarga del carrusel de datos mediante investigación y análisis de los protocolos pertenecientes a la estándar MPEG-2 ISO-13818-6: Extensiones para el DSM-CC, identificando cuáles son necesarios tomar en cuenta para la implementación del algoritmo.
- Estudiar y analizar la estructura del carrusel de objetos y de los diferentes tipos de mensajes (BIOP del inglés *Broadcast Interoperability Protocol*) que pueden encontrarse dentro del mismo carrusel.
- Diseñar el algoritmo para identificación y ordenamiento de objetos encontrados en el carrusel de objetos, mediante el análisis de parámetros específicos encontrados en las cabeceras BIOP, después de la descarga de datos.
- Elaborar el *software* extractor y constructor de datos mediante la implementación de los algoritmos diseñados, en funciones de MATLAB.
- Realizar una interfaz gráfica que permita al usuario seleccionar el TS y determinar el directorio donde se descargará y presentará los datos recuperados.
- Analizar tiempos de extracción de cada TS en función del número de bloques de control de descarga encontrados en un solo ciclo del carrusel de datos.

1.5. Estudio del Estado del Arte

Antes de que en el Ecuador se adapte oficialmente el estándar japonés-brasileño ISDB-Tb (año 2010) para Televisión Digital Terrestre, ya en instituciones como la Universidad de las Fuerzas Armadas ESPE se empezaron a realizar pruebas de generación de flujo de transporte MPEG2 TS con dicho estándar (Villamarin, 2013), incluyendo dentro del TS aplicaciones interactivas de GINGA NCL mediante la conversión de su contenido en un formato de carrusel de datos, con la ayuda de una herramienta de software gratuita. Las aplicaciones convertidas y multiplexadas dentro de TS, fueron ejecutadas correctamente por el decodificador receptor, por lo que la observación del contenido de la aplicación se logró solo mediante su ejecución.

Existen diversos *software* gratuitos como el *DemuxToy Lite* (Altais Digital Ltd., 2015) o pagados como el *TS Reader* (Cool.Stf., 2014), los cuales permiten analizar los TS y su contenido de forma detallada, mostrando principalmente las tablas de configuración PSI/SI y sus descriptores, dando al usuario la capacidad de saber qué tipo de servicios se encuentran en el TS y su respectiva configuración. En el caso del servicio de transporte de aplicaciones interactivas, los analizadores mencionados muestran información relevante como el PID de los paquetes de la sección de datos, identificándolo con el descriptor de *U-N DSM-CC Carrusel tipo B*, dentro de la PMT, y su configuración en la tabla de información de la aplicación (AIT del inglés *Application Information Table*). Funciones similares provee el *software* de (Benavides, 2015) específico para el estándar ISDB-T, que destaca por poseer una interfaz gráfica amigable con el usuario y su funcionamiento multiplataforma debido a su implementación en lenguaje *Java.*

En cuanto al control del DSM-CC a partir de la captura del TS, diversas arquitecturas son propuestas para mejorar el tiempo de descarga y extracción de datos como la de (Park, 2006), la cual permite una forma de almacenamiento rápido de la información en la memoria caché del decodificador, utilizando programación basada en prioridad por monitoreo de hilos, mientras que (Zhang, 2004) propone una modificación a la estructura tradicional de un carrusel de datos de modo que este transmita de una forma más sencilla la estructura jerárquica de archivos y directorios de la aplicación, y que de igual forma el receptor sea capaz de interpretar esta modificación para poder visualizar una mejora de rendimiento en la descarga de los datos. También (Guobin, 2006) propone otra arquitectura de software que permite un procesamiento más rápido de la descarga de información acorde a una selección previa del televidente.

Los trabajos citados anteriormente al igual que muchos basan sus objetivos en mejorar el rendimiento del sistema, disminuyendo tiempos de extracción por lo que la visualización y análisis del contenido se limita al uso de un decodificador especializado u otros dispositivos como el que propone (Washington, DC: U.S. Patente nº 6,931,198, 2005) donde se puede realizar la descarga del contenido de las

aplicaciones en un dispositivo de almacenamiento externo, para posterior análisis, pero esta forma aparte de ser patentada es mucho más compleja de implementar, razón por lo cual en el presente trabajo se propone generar un algoritmo de descarga, más sencillo y fácil de implementar en un software, con el objetivo de facilitar la tarea de extracción y reconstrucción de la información multimedia, cuando esta se transmite mediante en un carrusel de datos en el TS.

1.6. Descripción de Capítulos del Proyecto

El Capítulo 2 de este trabajo muestra la base teórica en la que se basa la trasmisión de datos de las aplicaciones interactivas en el TS, poniendo énfasis en detalles que fueron importantes para el desarrollo del software extractor, pero todo en concordancia con lo que dicta la norma MPEG-2 ISO-13818-6. En el Capítulo 3 se explica el método de extracción y construcción de datos, aplicado en el diseño del algoritmo implementado en el software, también una descripción básica de la herramienta de programación MATLAB y sus principales ventajas ante el desarrollo del proyecto. En el Capítulo 4 se presenta y analiza los resultados que se obtuvieron después del proceso de descarga y recuperación del contenido de las aplicaciones interactivas, mientras que las conclusiones y recomendaciones más representativas de este proyecto, se exponen en el Capítulo 5.

CAPÍTULO 2

2.1. Introducción

El avance que han realizado las nuevas tecnologías de comunicación ha permitido traer consigo el concepto y desarrollo de la Televisión Digital Terrestre (TDT), que ha convertido al telespectador pasivo en un usuario interactivo (Cho, 2008). Esto como consecuencia de la inclusión de aplicaciones interactivas dentro del sistema, las cuales cada vez más, son vistas como una nueva y potencial forma de hacer publicidad y comercio electrónico (Xiang, 2009), ya no solo como una forma de inclusión y equidad social de la población por parte de los gobiernos, como el caso de Ecuador (Ministerio de Telecomunicaciones y Sociedad de la Información, 2010) y otros países de América Latina (Dos Santos Jr, 2012). Esto produce que el contenido de información multimedia transmitido deba ser de alguna manera analizado, sea con fines regulatorios, de investigación o de comprobación del funcionamiento en cuanto a calidad de la programación de las aplicaciones interactivas, sin la necesidad de contar con un decodificador especial (Set-top box) que ejecute las aplicaciones y en consecuencia muestre su contenido.

La TDT transmite cada uno de sus diferentes servicios video, audio y datos, mediante el TS, el cual se caracteriza por realizar multiplexación de cada uno de estos servicios, en un solo flujo de datos, dividiendo la información en paquetes de 188 bytes los cuales se dividen 4 bytes de cabecera y 184 bytes de datos. La cabecera de los paquetes llevan su respectivo identificador de paquete (PID del inglés *Packet Identification*) que señala el tipo de datos que transporta los 184 bytes restantes, dependiendo de lo indicado en las tablas de información específica de programa (PSI del inglés *Program Specific Information*) o información de servicio (SI del inglés *Service Information*).

En el caso de la transmisión de datos de las aplicaciones interactivas, los paquetes transportan el directorio y contenido completo de las aplicaciones, a través de la conversión de estos a un formato de carrusel de datos (Villamarin, 2013) al descrito en la norma de MPEG-2 ISO-13818-6: Extensiones para el Comando y Control de

Almacenamiento de Medios Digital (DSM-CC del inglés Digital Storage Media Command and Control) (ISO/IEC, 1998).

2.2. Norma ISO 13818-6

La parte 6 de la norma ISO 13818, define un conjunto de protocolos funcionales para la especificación DSM-CC, que de forma combinada o individual pueden ser utilizados para proveer soporte a las tecnologías multimedia emergentes. En general DSM-CC provee la capacidad de explorar, seleccionar, descargar y controlar diferentes tipos de flujo de bits, a través del concepto "Sesión". La norma define la Sesión como el conjunto de recursos que sirve para proveer un servicio. En el caso de del TS del cual hace uso la TDT, los recursos para establecer una Sesión entre un Servidor y Cliente, pueden ser los identificadores de paquetes (PID), el ancho de banda de la red, etc (ISO/IEC, 1998).

En la norma se define 2 tipos de modelos, Usuario a Usuario (U-U ó *User to User*) y Usuario a Red (U-N ó *User to Network*), para conexiones punto a punto y punto a multipunto respectivamente. La figura 1 obtenida de (ISO/IEC, 1998) muestra los 2 modelos, donde el establecimiento de la Sesión y Administración de Recursos (SRM) solo se lleva a cabo en el modelo de conexión de Usuario a Red.

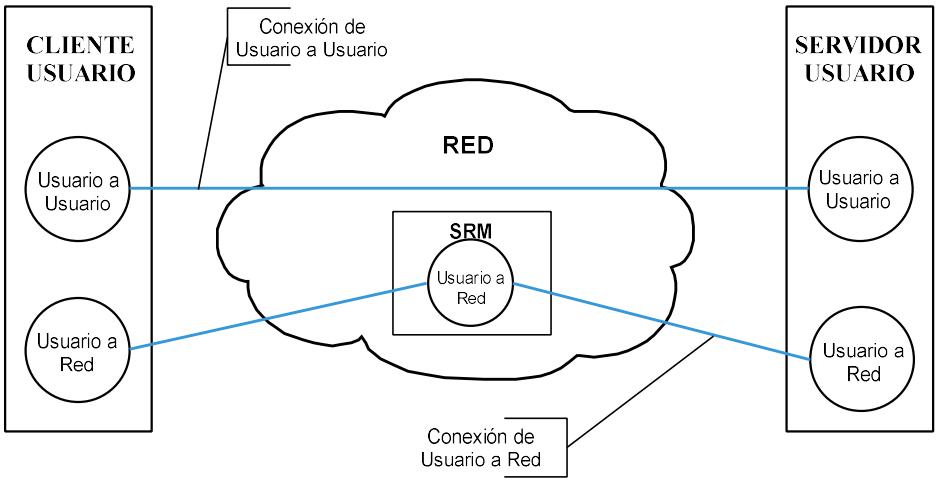


Figura 1. Modelo Cliente-Red-Servidor DSM-CC

El modelo U-N es asociado a una red de tipo *broadcast* unidireccional, como la que se tiene en la transmisión de un flujo de mensajes de datos desde un proveedor de servicio de transmisión (servidor) hacia un receptor *Set Top Box* (cliente), mientras que el modelo de U-U se asocia más bien con el canal de retorno de cliente a servidor. Por lo que para este trabajo solo es considerado el flujo de mensajes pertenecientes al modelo de U-N, debido a que son estos los que transportarán el contenido de las aplicaciones interactivas.

Los mensajes U-N deben ser transmitidos con un formato tal que los datos (de las aplicaciones interactivas) puedan ser extraídos y controlados por parte del receptor en cualquier momento de la transmisión. Esto se logra a través del perfil de protocolo de descarga, el cual considera el uso del concepto de "Carrusel de Datos", en conjunto con herramientas de apoyo para construcción de directorios y archivos como lo es el "Carrusel de Objetos" y para encapsulamiento dentro del TS las "Secciones DSM-CC". Estas herramientas y el concepto de Carrusel de datos son explicados a continuación.

2.3. Carrusel de Objetos

Las aplicaciones interactivas para TDT, son transmitidas con su directorio y archivos, entre los cuales se encuentra el código fuente de su programa principal, el cual generalmente funciona haciendo llamadas a imágenes, archivos de texto u otros ficheros multimedia que requiera para su ejecución, a partir del directorio donde estén ubicados, por lo que resulta importante no solo recuperar el contenido de los archivos en sí, sino también la ubicación u ordenamiento archivos en relación a su directorio raíz. De esto se encarga el carrusel de objetos que provee un sistema de mensajes "*Broadcast Interoperability Protocol*" (BIOP), el cual se encarga de proveer el sistema de recuperación de archivos con su directorio (ETSI TR., 2003). Cada mensaje BIOP representa a un objeto en particular y cuyos tipos se describen a continuación

2.3.1. Tipos de Objeto Soportados

La norma ISO 13818-6 define 5 tipos de objetos soportados en DSM-CC, los cuales son mostrados en la tabla 1. Cada tipo de objeto cuenta con su identificador (nombre al que se refiere comúnmente dentro de la norma) y su alias (nombre con el que cada objeto se describe dentro su estructura).

Tabla 1.
Tipos de objeto

Tipo de Objeto	ID de Tipo	Alias
Directorio	*Directory*	'dir'
Archivo	*File*	'fil'
Puerta de Servicio	*Service Gateway*	'srg'
Flujo	*Stream*	'str'
Evento de Flujo	*Stream Event*	'ste'

Cada tipo de objeto es definido y explicado a continuación.

2.3.1.1 Objeto Directorio

El objeto Directorio o *Directory*, se define como una lista o grupo de enlaces que relaciona el nombre de objeto con un directorio en particular. Además cada enlace puede contener atributos relacionados con el objeto para facilitar su navegación entre directorios (ETSI TR., 2003). Actualmente el atributo que más se encuentra y se utiliza es el tamaño del contenido (*contentSize*). Cada enlace es representado con una clave de objeto (*objectKey*), la cual es muy importante para la reconstrucción del directorio completo de la aplicación. La estructura y sintaxis del mensaje BIOP de tipo *Directory* puede verse en el Anexo 1.

2.3.1.2 Objeto Archivo

El objeto Archivo o *File* es el principal encargado de contener los datos de los archivos, independientemente del formato que estos posean. También a este se le

asigna una clave de objeto con la cual se crea un enlace con su respectivo directorio. La estructura y sintaxis del mensaje BIOP de tipo *File* puede verse en el Anexo 1.

2.3.1.3 Objeto Puerta de Servicio

El objeto Puerta de Servicio o *Service Gateway* al igual que el *Directory*, representa una lista de enlaces entre nombres y directorio de objetos, con la particularidad de que este representa al directorio raíz de la aplicación y no cuenta con un nombre definido como los demás directorios. La estructura y sintaxis del mensaje BIOP de tipo *ServiceGateway* puede verse en el Anexo 1.

2.3.1.4 Objeto Flujo

El objeto Flujo o *Stream*, contiene un grupo de identificadores a los cuales la norma denomina como *Taps*. Cada *Tap* se relaciona con un flujo elemental o con un programa completo del TS. Además los identificadores también pueden contener información de control de flujo (ETSI TR., 2003). En la práctica actualmente este tipo de objeto no es muy utilizado para el tipo de mensajes U-N, y tampoco resultan de importancia para los objetivos de este trabajo, por lo que no fueron considerados dentro de la elaboración del algoritmo de extracción.

2.3.1.5 Objeto Flujo de Evento

El objeto Flujo de Evento o Stream Event es muy similar al objeto *Stream*, con la diferencia de que este añade otros atributos como lista de eventos (*EventList* y *EventIds*) los cuales contienen una lista de nombres de eventos del DSM-CC (ETSI TR., 2003). Este tipo de objeto al igual que el de *Stream* no es considerado dentro del algoritmo de extracción.

2.3.2. Sistema de Encapsulamiento y Transmisión de Objetos

Los mensajes BIOP son parte de un sistema de encapsulación y transmisión de los objetos anteriormente descritos, de tal forma que estos puedan ser recuperados junto con su respectiva organización jerárquica de archivos y directorios. La figura 2

muestra un ejemplo muy sencillo de este sistema, el cual se basa en la asignación de claves de objetos a cada archivo y carpeta para después encapsular la información dentro de mensajes BIOP de tipo *file* (*fil*), *directory* (*dir*) y *service gateway* (*srg*), cada uno con su respectiva cabecera la cual contiene información del tipo de mensaje, tamaño y sobre todo la clave de objeto asignada.

Como se mencionó anteriormente los mensajes tipo *fil* son los encargados de transportar el contenido de los archivos, mas no su nombre, ya que de esto se encargarán los mensajes *dir* y *srg*. Cabe mencionar que la estructura de los mensajes *dir* y *srg* son exactamente iguales con la diferencia que *srg* generalmente se le asigna la clave de objeto cero (0) y será único en el carrusel de objetos ya que representa al directorio raíz, mientras que los tipo *dir* puede haber tantos como el número de total de carpetas en el directorio (ETSI TR., 2003).

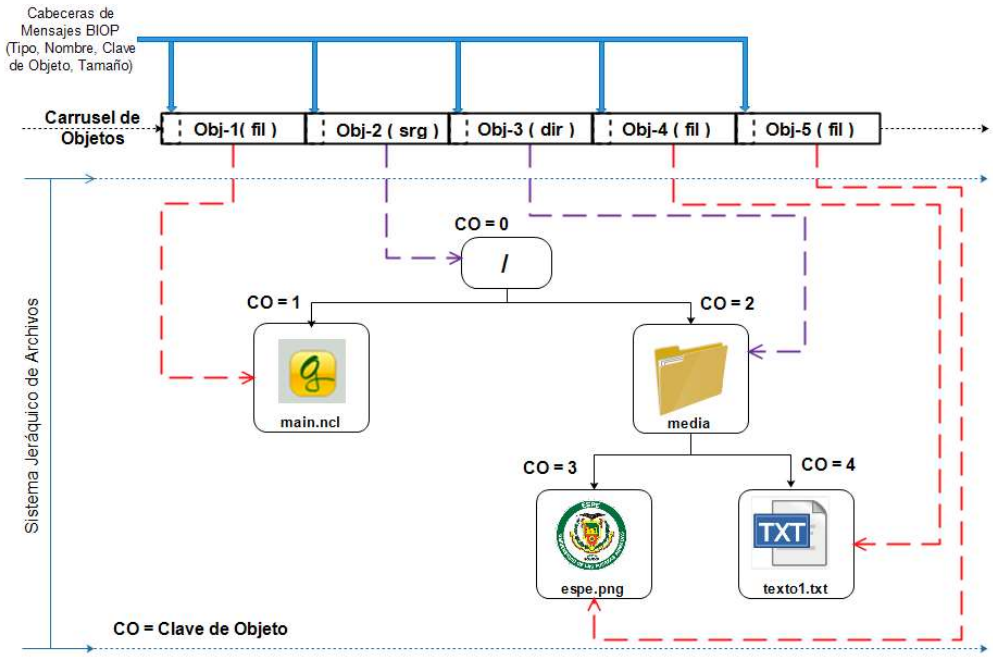


Figura 2. Encapsulación de una aplicación GINGA-NCL en carrusel de objetos

Los mensajes *dir* y *srg* por lo tanto son los encargados de describir los archivos y carpetas que contiene un directorio mediante una serie de enlaces que identifican su nombre, tipo, tamaño (solo de archivos) y en especial las claves de objetos que se les fue asignado. La tabla 2 muestra los enlaces que deberían tener los mensajes *dir* y *srg*,

haciendo referencia al ejemplo de la figura 2. Los nombres, tipos y claves de objetos asociados son los que sirven para relacionar un directorio con los archivos o carpetas que estos deben contener

Tabla 2.
Enlaces de componentes asociados al directorio (*dir* o *srg*)

Clave Objeto Directorio	Nombre Objeto Asociado	Tipo Objeto Asociado	Clave Objeto Asociado
0	main.ncl	fil	1
	media	dir	2
2	espe.png	fil	3
	texto1.txt	fil	4

2.4. Carrusel de Datos

El objetivo del carrusel de datos es proveer un formato a los datos para facilitar la transmisión a través de la red, por lo que después de haber convertido la información en carrusel de objetos, este es dividido en módulos, y a su vez cada módulo se subdividirá en un determinado número de bloques de descarga.

2.4.1. Mensajes de Descarga

Existen 3 tipos de bloques o mensajes de descarga dentro de un carrusel de datos: El *Download Data Block* (DDB), el *Download Info Indication* (DII) y el *Download Server Initiate* (DSI). Los DDB son los encargados de encapsular al carrusel de objetos, es decir que en general transportan los datos de la aplicación. Los DII y DSI por otro lado constituyen mensajes de control de descarga, con la particularidad de que un DII tiene la función de describir a un solo grupo de módulos, mientras que el DSI describe a un supergrupo (dos o más grupos) (ETSI TR., 2003). La estructura y sintaxis de los mensajes de descarga DDB, DII y DSI puede verse en el Anexo 2.

2.4.2. Transmisión cíclica del carrusel.

El tipo de trasmisión del carrusel de datos hacia la red es de naturaleza cíclica, es decir que este se encontrará repetido varias veces a lo largo de todo el TS, para permitir

al cliente la descarga de la información en cualquier momento. Un ejemplo de esto se muestra en la figura 3 obtenida de (ETSI TR., 2003), donde se tiene un carrusel de datos con 3 módulos (M1, M2 y M3) en un bucle que se repite cíclicamente y donde cada módulo es dividido en un cierto número de bloques DDB dependiendo del tamaño del módulo. Además se tiene la presencia de bloques de control de descarga como el DII o el DSI, los cuales describen al grupo de módulos en cuestión.

Cabe destacar que aun cuando en el ejemplo de la figura 3 se tiene una sola vez la inclusión de la DII y DSI dentro del bucle, la norma ISO/IEC 13818-6 no restringe en lo absoluto la posición, las veces o la frecuencia con la que se puedan introducir estos bloques dentro de un mismo ciclo, con el objetivo de generar el carrusel que mejor se adapte con las necesidades o características del cliente.

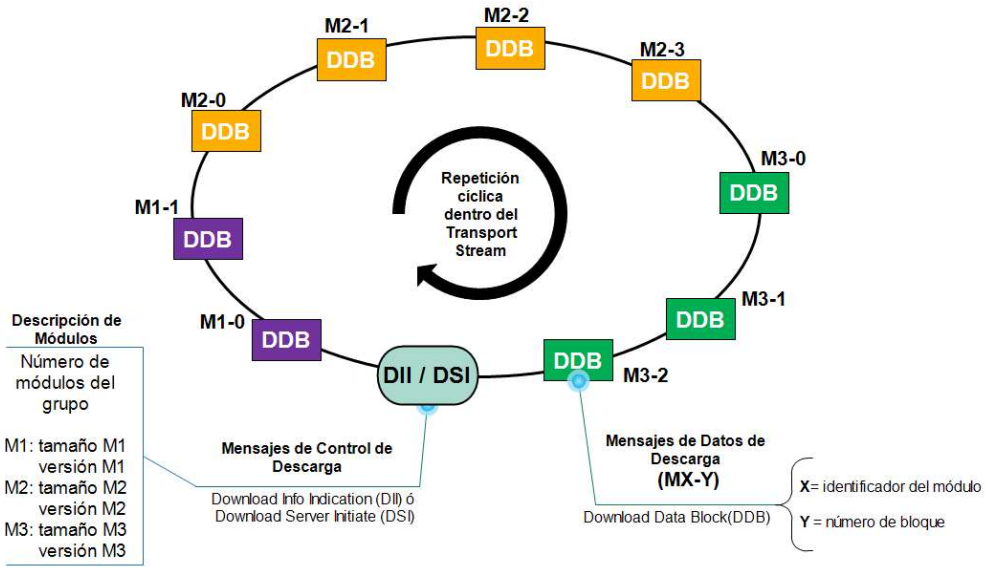


Figura 3. Transmisión cíclica de información en un carrusel de datos

2.4.3. Carrusel de datos de una y dos capas: uso y diferencias

Existen dos tipos de carrusel de datos: De una capa y de dos capas. El carrusel de una capa se da en el caso donde se tiene un solo grupo de módulos, que es descrito por un único DII, mientras que el de dos capas tiene dos o más grupos (supergrupo), descritos por un DII diferente en cada grupo. El conjunto DII de un carrusel de datos de dos capas es descrito por un único DSI, conformando así la descripción de un

supergrupo. La comparación del carrusel de una capa y de dos capas se muestra en la figura 4 obtenida de (ETSI EN, 2004).

El carrusel de dos capas es utilizado solo en casos donde el carrusel de datos consta de un número grande de módulos (mayor a 65.535), razón por la cual su descripción es demasiado grande para un solo mensaje DII o cuando se tiene variabilidad en cuanto a la versión de una aplicación transmitida, respecto al perfil que maneja el receptor (ETSI TR., 2003). Entonces dado que en este trabajo solo utilizamos como base la transmisión de aplicaciones en una única versión y además por su cantidad limitada de elementos, se generarán un número pequeño de módulos para el carrusel de datos, resultando efectivo la utilización de un solo DII para contener toda la descripción del grupo, por lo que se planteó solo el uso de concepto de carrusel de datos de una capa, para la elaboración del algoritmo de extracción.

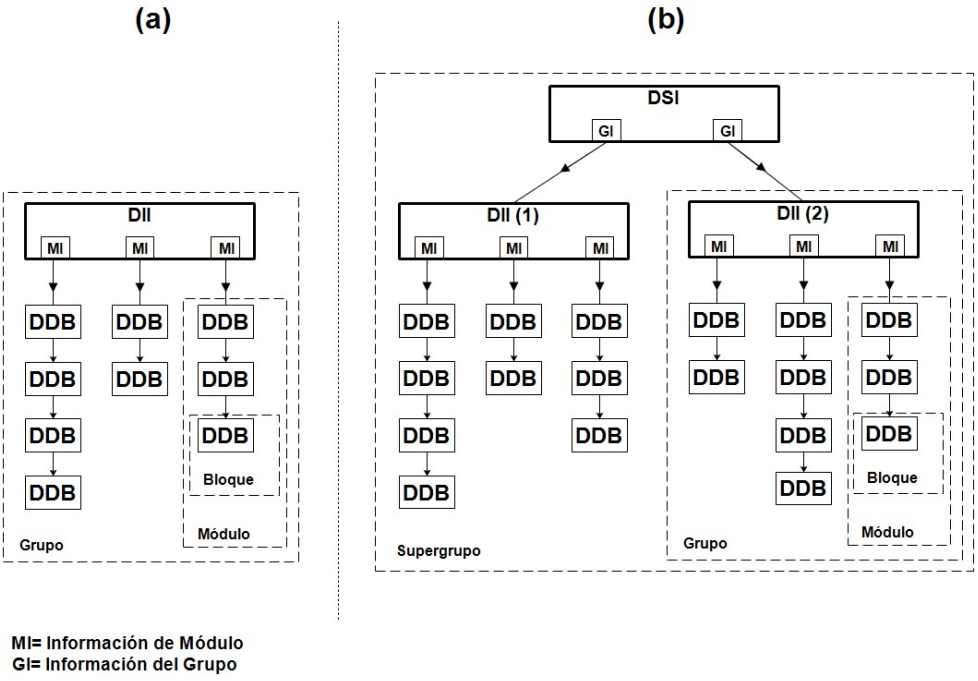


Figura 4. Carrusel de datos: (a) una capa, (b) dos capas

2.5. Secciones DSM-CC

Las secciones DSM-CC constituyen una forma que la norma ISO/IEC 13818-1 (ISO/IEC, 2015) utiliza para ensamblar los paquetes del TS dentro del carrusel de

datos. Por lo que esto permite la localización de bloques o módulos requeridos a partir de los paquetes que posean el PID correspondiente al *stream* de tipo U-N DSM-CC. La tabla 3 muestra la estructura de las secciones DSM-CC con respecto a los bloques DDB, DII o DSI del carrusel de datos. El DII y DSI corresponden al *User Network Message* con un *table id* de 0x3B, mientras que el DDB al *Download Data Message* con *table id* de 0x3C. Existen otros valores de *table id* aparte del 0x3B o 0x3C, que son para bloques de bytes de datos privados, listas de descriptores DSM-CC, etc, sin embargo estos no son de relevancia para los objetivos de este trabajo por lo que no son tomados en cuenta.

Tabla 3.
Estructura de sección DSM-CC

Sintaxis	No. de Bits
DSMCC section(){	
table_id	8
section_syntax_indicator	1
reserved	1
dsmcc_section_length	12
table_id_extension	16
reserved	2
versión_number	5
current_next_indicatot	1
section_number	8
last_section_number	8
if(table id==3B){	
userNetworkMessage()	
}	
else if (table id==3C){	
downloadDataMessage()	
}	
if(section syntax indicator==’0’){	
checksum	32

Continua ➔

}	
else{	
CRC_32 }}	32

El tamaño máximo de una sección es de 4096 bytes (ISO/IEC, 2015), por lo que en el campo *dsmcc section length* de la tabla 3 es común encontrar el valor de 4093 bytes, considerando que esta longitud se toma a partir del campo *table id ex* hasta el final de la sección incluyendo el *checksum* o el CRC-32. Sin embargo este valor de longitud de sección casi siempre es menor de 4093 para la última sección de un cada módulo. Es decir usualmente para cuando en una sección el valor del campo *section number* y *last section number* son iguales.

Según el *table id* que se presente habrá valores comunes de ciertos parámetros de la sección DSM-CC, los cuales son mostrados en la tabla 4, donde en lo que respecta al DDB, los valores son los mismos vistos en el carrusel de datos de la figura 3 como el *module ID* (X) y el *block Number* (Y). En lo que refiere al DSI y el DII la tabla 3 muestra valores con los cuales se puede diferenciar uno de otro, tomando en cuenta que los dos poseen el mismo *table id* de 0x3B.

Tabla 4.
Codificación de las secciones DSM-CC

Mensaje	DDB	DII	DSI
table_id	0x3C	0x3B	0x3B
table_id_extension	module Id	0x0002-0xFFFF	0x0000-0x0001
versión_number	module Version	0x00	0x00
section_number	blockNumber(0x00-0xFF)	0x00	0x00
last_section_number	Max(section number)	0x00	0x00

2.6. Encapsulamiento y mapeo de secciones DSM-CC en el TS

El inicio o final de una sección DSM-CC no es estrictamente marcado por el inicio o final de un paquete de transporte, por lo que MPEG-2 TS provee un byte de control llamado *poiter field* con el que se puede mapear el inicio de una nueva sección a partir

de la cabecera del paquete TS. El *poiter field* y su funcionamiento son mostrados en la figura 5. Cuando el indicador de arranque (10mo bit de la cabecera del paquete TS) se encuentra puesto en '1', significa que primer byte después de la cabecera corresponderá al *poiter field*. El valor máximo del *poiter field* será de 182 (Ether Guide Systems, 2016).

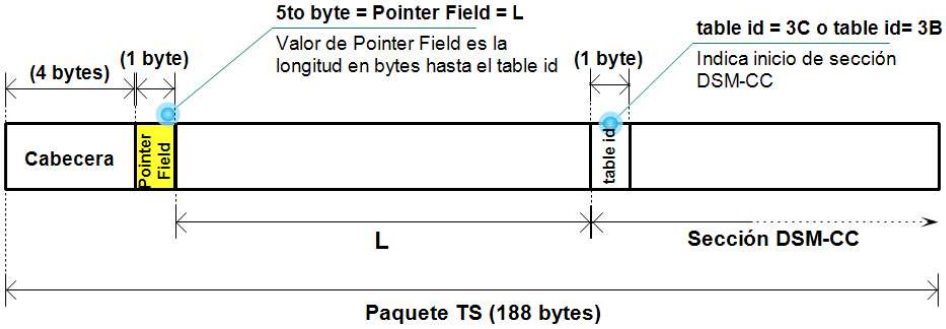


Figura 5. *Pointer field* en paquete TS para localización del *table id*

No puede existir más de un *pointer field* dentro de un mismo paquete, es decir que si una sección es tan pequeña que inicia y termina en el mismo paquete, se procede a rellenar con bytes (0xFF) desde el final de dicha sección hasta el final del paquete, quedando el inicio de una nueva sección en el siguiente paquete con un valor de *pointer field* igual a cero. El relleno también puede ser identificado con el control de campo de adaptación de la cabecera del paquete TS (ABNT NBR, 2009).

2.7. Localización de paquetes DSM-CC en el TS

Como se mostró en la figura 5, el tamaño de un paquete del TS es de 188 bytes, de los cuales 184 bytes corresponden a la carga útil, mientras que los 4 bytes iniciales constituyen la cabecera que contiene información que determina el tipo o función de un paquete.

La localización del valor de PID correspondiente a los mensajes U-N DSM-CC, se lleva a cabo mediante el análisis de dos tablas PSI: La tabla de asociación de programas (PAT del inglés *Program Association Table*) y la PMT. La PAT que tiene el PID reservado igual cero (0) y es la encargada de identificar que programas contiene

el TS y sus respectivos PID's, mientras que la PMT identifica los servicios que posee cada programa (incluyendo las aplicaciones interactivas), mediante la utilización de descriptores. Cada descriptor cuenta con un identificador de tipo de flujo (*Stream type identifier*) que determina qué tipo de servicio representa el descriptor. En el caso del servicio de mensajes U-N DSM-CC que contiene los datos de las aplicaciones interactivas, el descriptor que contiene su PID debe ser localizado con un identificador de flujo de valor 0x0B (ABNT NBR, 2009). La figura 6 muestra los valores del descriptor del flujo para el servicio de mensajes U-N DSM-CC.

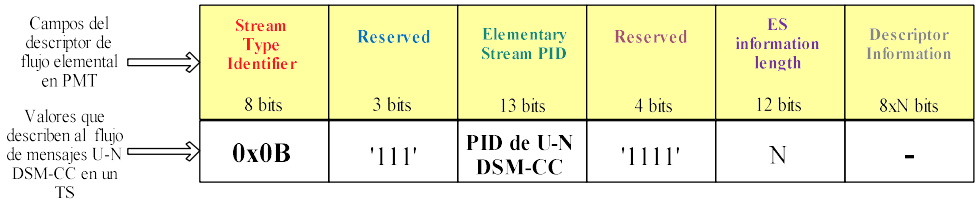


Figura 6. Descriptor de flujo en PMT para el DSM-CC

Si la PMT de un programa en específico del TS no posee un descriptor con identificador de flujo de 0x0B, significa que este no posee o no está asociado a ninguna aplicación interactiva.

CAPÍTULO 3

3.1. Elaboración del algoritmo extractor.

La encapsulación de datos de aplicaciones interactivas dentro del TS (de MPEG-2) para su transmisión en una red *broadcast* puede ser expresada como un modelo por capas, muy similar a los modelos de red OSI o TCP/IP, en donde cada capa del modelo añade su respectiva cabecera a un segmento de datos y que cumple con una determinada función. La figura 7 muestra la encapsulación de datos expresada en un modelo por capas, la cual permite entender el proceso de desencapsulación que debe seguir el algoritmo extractor para cumplir con su objetivo de manera eficaz.

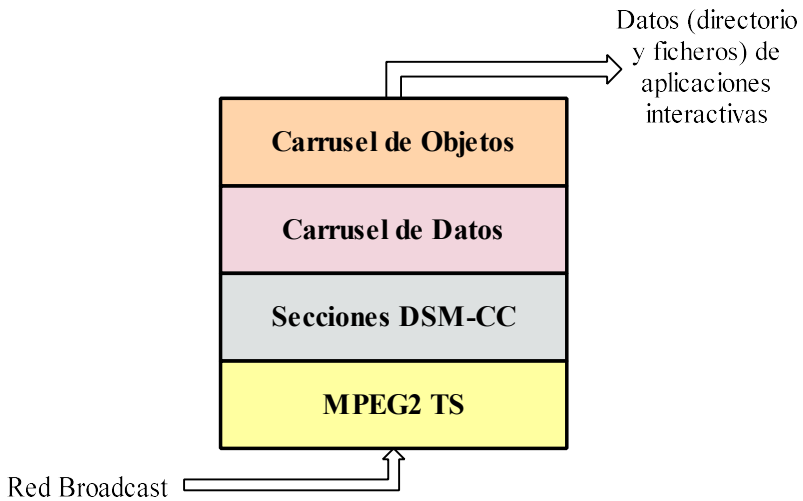


Figura 7. Modelo por capas de encapsulamiento de datos en el TS

Dado que este trabajo se centra en la extracción de datos a partir de un TS recuperado de la señal *broadcast* de TDT, se determinó que la elaboración del algoritmo extractor se debía basar en 3 etapas principales, refiriéndose a las desencapsulación de secciones DSM-CC, carrusel datos y carrusel de objetos. Cabe mencionar que para explicar el proceso de desencapsulación se utilizará como base un TS de prueba llamado *celinaginga.ts*, el cual posee sección de datos DSM-CC que contienen una aplicación interactiva de GINGA-NCL.

3.1.1. Desencapsulación de secciones DSM-CC con *table id* 0x3C

La desencapsulación de secciones DSM-CC empieza a partir de la identificación del PID de los paquetes que contienen los mensajes U-N DSM-CC, para lo cual se realiza un análisis de las tablas PAT y PMT. Un ejemplo de este análisis se muestra en la figura 8, donde se utilizó el *software* de (Benavides, 2015) para poder visualizar los bytes de los paquetes del TS *celinaginga.ts* expresados en dígitos hexadecimales. Como se observa en este caso particular después del análisis respectivo del descriptor del flujo de datos 0x0B, se encontró que el PID 2004 corresponde al identificador de paquetes que contiene los mensajes DSM-CC requeridos.

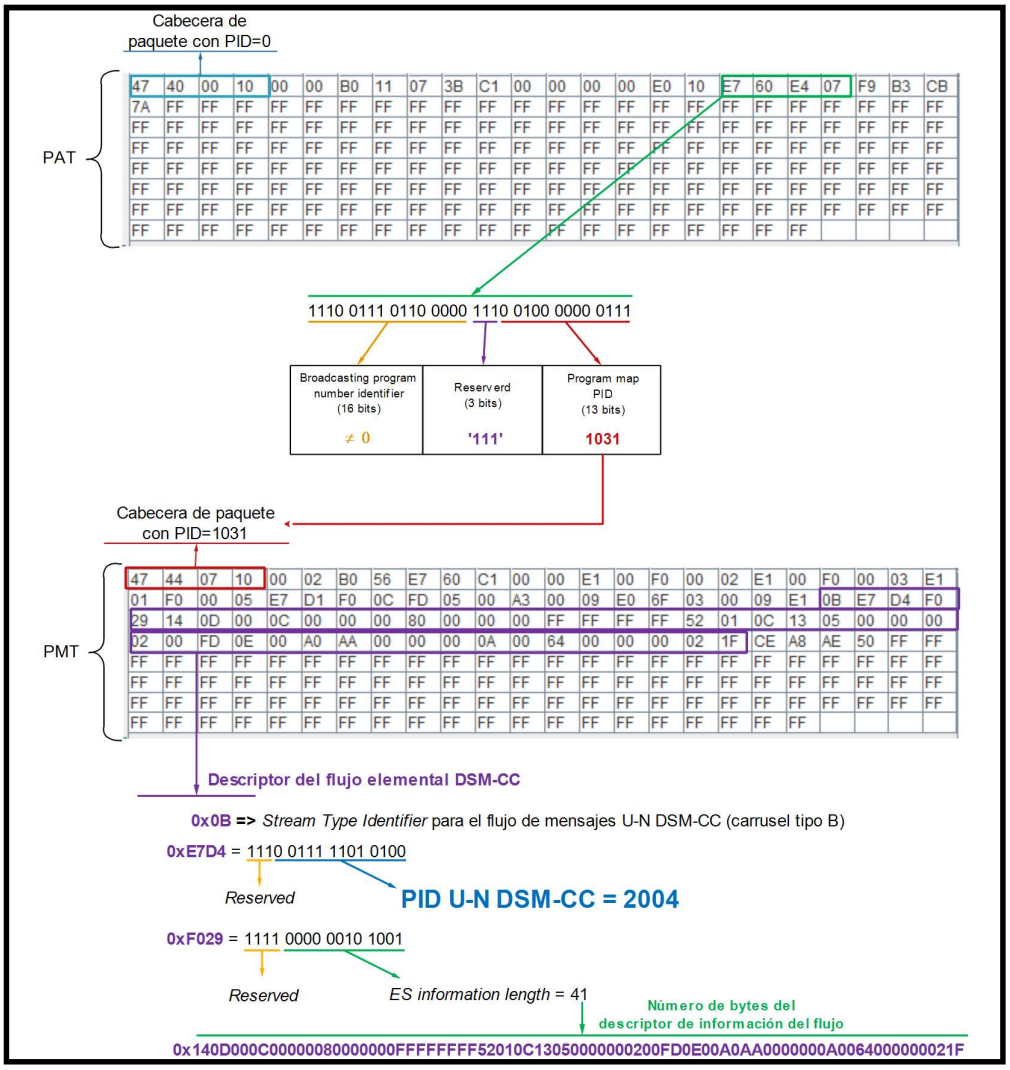


Figura 8. Análisis de PAT y PMT para localizar el PID de mensajes DSM-CC

Una vez identificado el PID de los paquetes DSM-CC se extrae las secciones DSM-CC utilizando el *pointer field* para localizar el inicio y final de cada sección, además es necesario analizar ciertos parámetros de la cabecera de cada sección (mostrados en la tabla 3) para identificar valores que determinen el tipo, tamaño, función o módulo al que pertenece la sección. La figura 9 muestra un ejemplo de la desencapsulación de la primera sección del módulo 1 del TS de prueba.

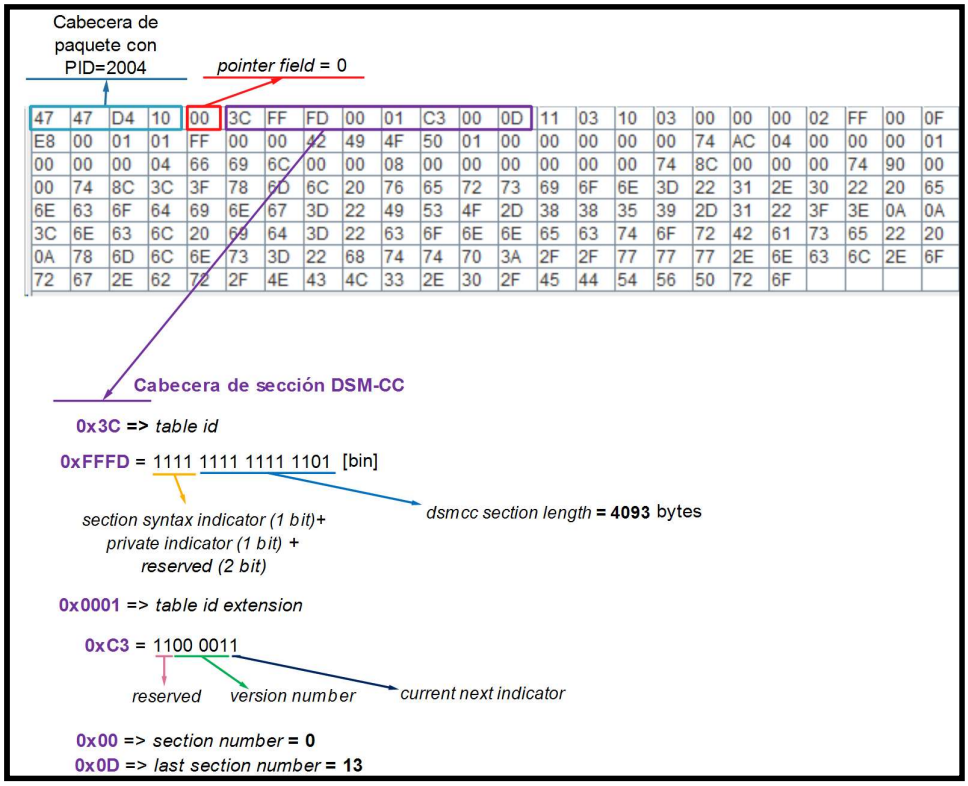


Figura 9. Análisis de inicio de sección DSM-CC

El paquete analizado en la figura 9 con PID igual a 2004 contiene la parte inicial de una sección DSM-CC. La ubicación del *table id* de la sección dentro del paquete en cuestión es marcada por el *pointer field*, que al ser de valor cero (0) indica que la distancia en bytes de este puntero hasta el *table id* es nula, es decir que el *table id* se encuentra a continuación del *pointer field* identificando con 0x3C a una sección que contiene un bloque de datos de descarga (DDB).

El bit de *section syntax indicator* se encuentra puesto en '1' por lo que indica que el método de control de errores en la sección será a través del código de verificación por redundancia cíclica CRC-32 y además que el 4to y 5to byte de la cabecera de la sección corresponderá al campo *table id extensión* (ETSI TR., 2003). Según lo mostrado en la tabla 4, ya que la sección tiene el *table id* 0x3C, el campo *table id extensión* corresponde al identificador de módulo (*module id*) al que pertenece la sección, lo cual indica que en este caso en particular pertenece al módulo 1 (0x0001) del carrusel de datos.

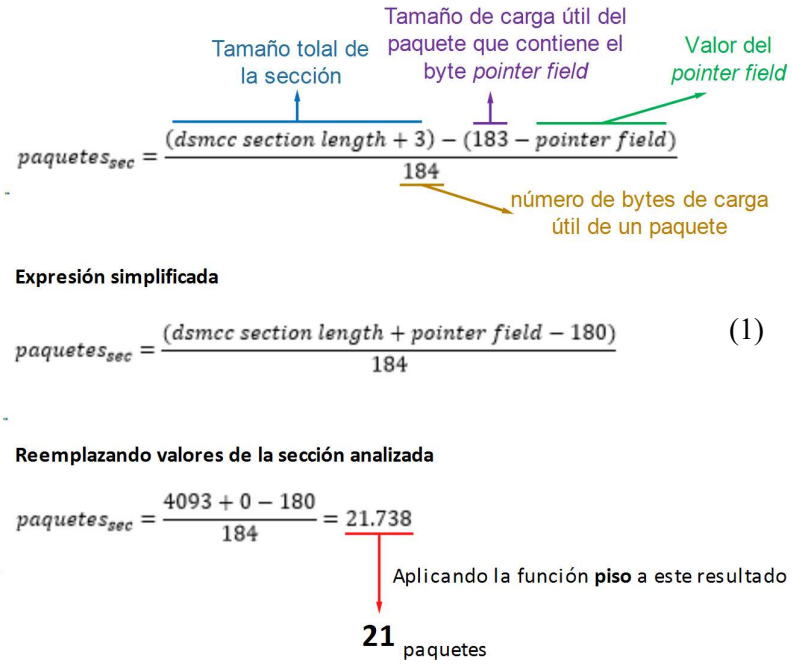
La longitud o tamaño de la sección es dada por el campo *dsmcc section length* que para este caso tiene un valor de 4093 bytes. Este tamaño se cuenta inmediatamente después de este campo, es decir a partir del cuarto byte de la sección, por lo que sumando los 3 bytes iniciales de la cabecera da como resultado 4096 bytes de tamaño total de la sección analizada. Es necesario recordar que un grupo de secciones (que contienen únicamente DDB's) puede representar a un módulo en específico del carrusel de datos y que usualmente cada sección mantiene un tamaño máximo de 4096 bytes (4 kbytes) a excepción de la última sección de cada módulo que por lo general tendrá un tamaño inferior al mencionado, en cuyo caso existe ciertas consideraciones especiales, la cuales se deben tomar en cuenta para su correcta extracción.

El campo *versión number* (en el caso de una sección que contiene un DDB) de 5 bits identifica la versión del módulo (*module version*) al que pertenece. Según la norma (ETSI TR., 2003) la versión del módulo puede ser visto como un identificador de sub-tabla, cuyo valor se incrementa en cuanto existe un cambio en un módulo. Los bloques de control de descarga (DII) serán los encargados de actualizar la información de los módulos respecto al cambio de versión de un módulo en específico. El bit *current next indicator* se encuentra puesto en '1' indicando que el uso de la sub-tabla esta activa. La norma (ABNT NBR, 2008) indica que cuando se trate de una sección con *table id* 0x3B o 0x3C el *current next indicator* siempre se configurará en estado '1'.

El campo *section number* con valor cero (0) de la figura 9 indica que es la primera sección del módulo y este valor se incrementará sección a sección, hasta alcanzar o

igualar el valor del campo *last section number* que para este caso es 13. Usualmente el valor del campo *last section number* mas uno, representa el número total de secciones o bloques en que se divide un módulo, por lo que para el caso del módulo analizado sería un total de 14 secciones. Sin embargo, esto solo es aplicable para cuando el módulo no es demasiado grande como para tener que dividirse en más de 255 secciones, el cual es el número máximo que un campo de 8 bits puede alcanzar.

Por otro lado, es necesario identificar también el final de una sección o a su vez el inicio de una sección posterior, para lo cual si bien es cierto se puede contar con la ayuda del bit de indicador de arranque (*payload unit start indicator*) ubicado en la cabecera del paquete, resulta mejor utilizar el campo *dsmcc section length* de la cabecera de la sección, ya que este permite a través de un cálculo estimar el número de paquetes intermedios entre los paquetes que contienen el inicio y final de la sección, lo cual contribuye a la disminución del costo computacional al implementar un *software* con un algoritmo extractor mediante este método, debido a que así no será necesario verificar el estado del indicador de arranque de la cabecera para todos los paquetes. Un ejemplo de este cálculo y su explicación se muestra continuación mediante la ecuación 1.



En el ejemplo de cálculo anterior se obtuvieron 21 paquetes después de aplicar la función *piso*, la cual toma únicamente la parte entera del resultado de *paquetes_sec*. Por lo tanto 21 será el número de paquetes que se encuentra entre el inicio y final de la sección para este caso en particular. La figura 10 muestra la utilización del valor 21 calculado. El paquete 1 es el mismo que se analizó en la figura 9, donde inicia la primera sección de las 14 que componen el módulo es decir la sección DSM-CC 1/14. Después se toman los 21 paquetes, es decir del paquete 2 al 22 el cual el grupo de paquetes que únicamente contienen la carga útil de la sección y que por consiguiente no poseen el campo *pointer field*. Cabe mencionar que el cálculo de *paquetes_sec* usualmente variará entre 21 y 22 paquetes (dependiendo del valor del *pointer field*) siempre y cuando la sección tenga el tamaño máximo de 4096 bytes.

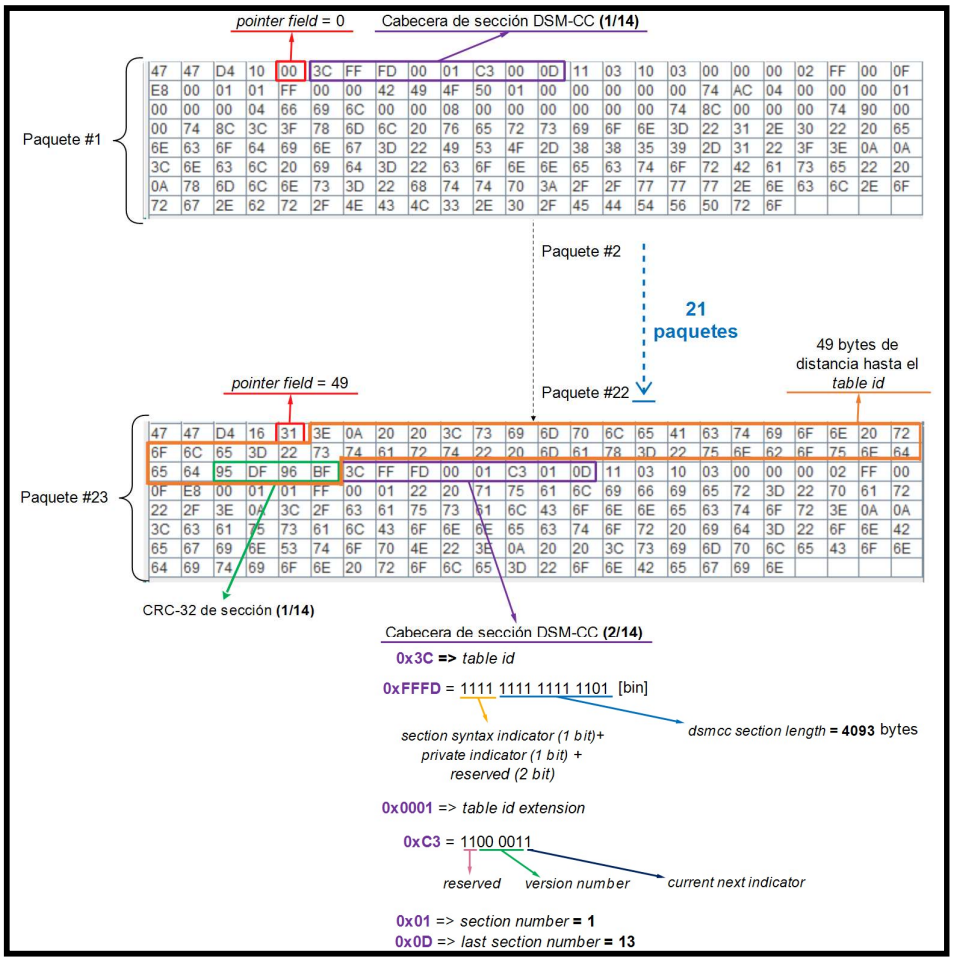


Figura 10. Uso del valor *paquetes_sec* en función del tamaño de la sección

A continuación en el paquete 23 con ayuda del *pointer field* (0x31=49 bytes) se puede localizar tanto el fin como el inicio de la segunda sección del módulo es decir la 2/14. El fin de la sección 1/14 es marcado por los bytes correspondientes al CRC-32 (4 bytes) mientras que el inicio de la siguiente sección es marcado como siempre por el *table id* y los demás campos que componen la cabecera de la sección. Como se observa el único campo de la cabecera que varía entre la primera y segunda sección es el *section number*, demostrando la continuidad y el orden que tienen las secciones que componen un módulo. Es importante recordar que entre secciones pertenecientes a un módulo, es posible encontrar una o varias secciones que proveen control para extracción o descarga, las cuales como se expuso anteriormente tendrán un *table id* de 0x3B. La localización y uso de las secciones de control se explicaran posteriormente.

La desencapsulación de secciones de secciones DSM-CC con *table id* de 0x3B trata entonces de identificar el inicio y final de cada sección así como el uso de los campos de sus cabeceras, tales como el *dsmcc section length*, *table id extensión* (*module id*), *section number* y *last section number* para así extraer los bloques de descarga DDB en un solo fichero temporal que contenga el carrusel de datos como tal.

3.1.2. Localización y uso de secciones DSM-CC con *table id* 0x3B

Las secciones DSM-CC con *table id* 0x3B son aquellas que contienen bloques de control de descarga tales DII o DSI. Su localización es igual a la de las secciones DSM-CC con *table id* 0x3C, la cual se realiza en base al campo *pointer field* y su *table id* pero su uso dentro del algoritmo propuesto no se basa en su extracción sino más bien en la identificación de parámetros que proveen el bloque de control mediante los campos dentro de su estructura. Es decir no solo se tomará en cuenta los parámetros de la cabecera de la sección sino también aquellos que contiene el bloque de control como tal. La figura 11 muestra el análisis que considera el algoritmo respecto a la sección que contiene un bloque de control para lograr una correcta extracción de los bloque de datos de descarga.

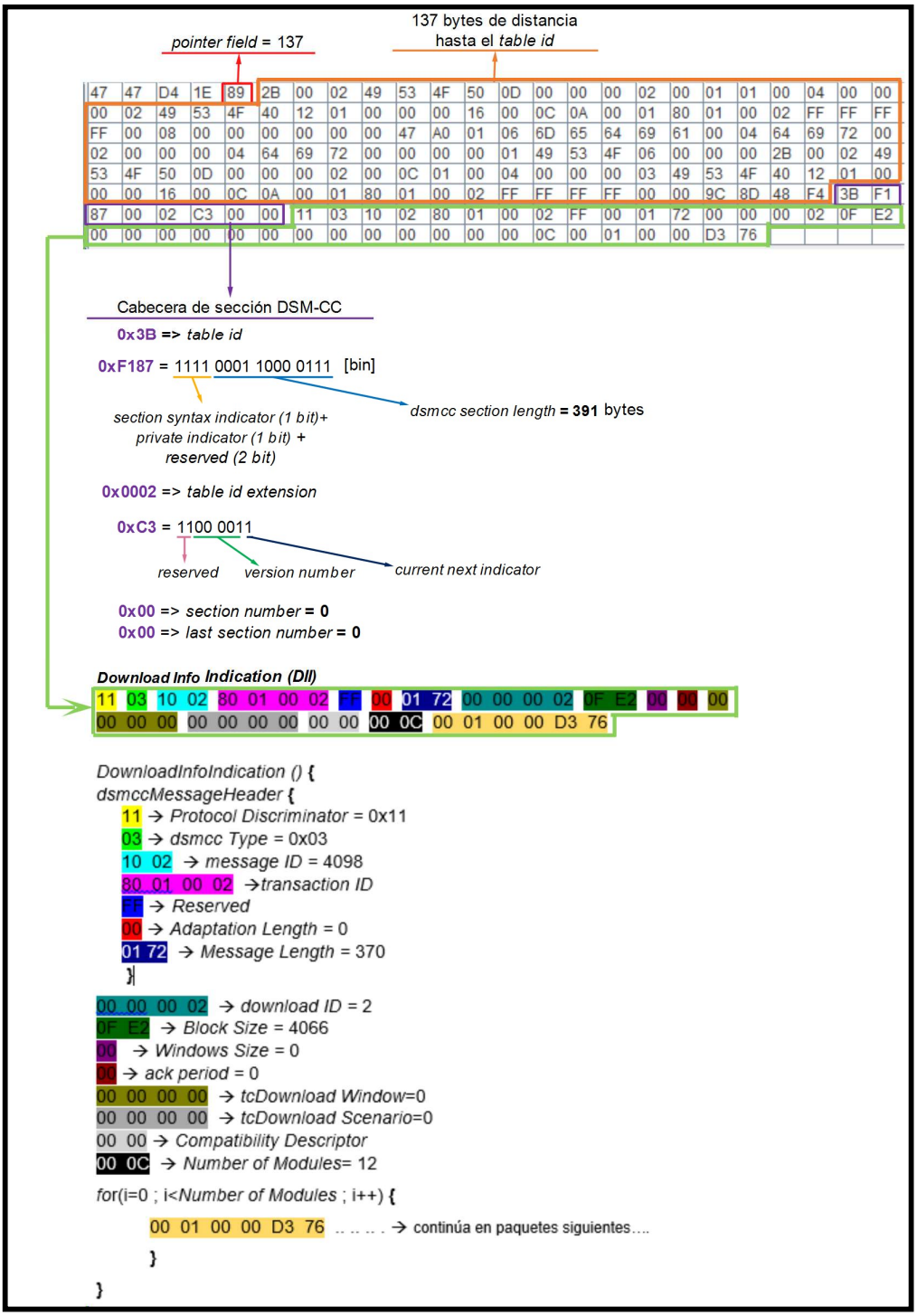


Figura 11. Análisis de cabecera de sección y bloque de Control

Dentro del análisis de la cabecera de la sección de la figura 11 se identifica mediante el *table id* 0x3B, que esta corresponde a una sección que contiene un bloque

de control y mediante el *table id extensión,* acorde con la tabla 4 se determina que este bloque de control es un DII. Después en base a la estructura de los bloques DII se hace el análisis e identificación de campos importantes para el diseño del algoritmo como lo es *dsmcc Type* igual a 0x03 que confirma que los bloques son del tipo de mensajes U-N y el *Number of Modules* que sirve para conocer cuántos módulos componen un solo ciclo del carrusel de datos, siendo esto importante tanto para conocer el número necesario de módulos que se deben descargar así como también para determinar en qué momento se debe finalizar la extracción de las secciones DSM-CC con *table id* 0x3C.

La descripción de las características de cada módulo (identificador, tamaño, versión y bytes de información) se lleva cabo dentro del ciclo de repetición (*for*) de la estructura del DII, el cual actúa en función del campo *Number of Modules*, sin embargo esta descripción no es necesaria tomarla en cuenta debido a que estos datos no son de relevancia para el control de descarga que se propone en el algoritmo extractor de secciones, por lo que no es importante realizar un análisis detallado de estos bytes.

3.1.3. Consideraciones especiales en el uso del *pointer field*

El campo *pointer field* es útil para lograr una correcta localización y extracción de las secciones DSM-CC, sin embargo existen casos especiales en las que el *pointer field* debe funcionar en conjunto con bytes de relleno (0xFF), para cuando se presentan ciertas condiciones o particularidades en paquetes que contienen el final de la última sección que compone un módulo. Es decir, aquella sección que por lo general tiene un tamaño inferior a 4096 bytes.

3.1.3.1 Caso 1: 183 bytes de datos en paquete final de una sección

El valor que contenga el campo *pointer field* también puede ser interpretado como la cantidad de bytes del final de una sección (incluyendo el CRC-32) que estarán ubicados en el paquete que contiene dicho *pointer field*. Esto limitaría en teoría al *pointer field* entre un valor de 0 a 183 siendo este último valor el número máximo de bytes de carga útil de un paquete que contiene un *pointer field*. Sin embargo la norma

ISO 13818-1 limita al *pointer field* entre un valor de 0 a 182 generando un caso especial para cuando el número de bytes de la parte final de la sección (ubicados en el último paquete que la contiene) coincide que es igual a 183. La figura 12 muestra un ejemplo del porqué no se permite un *pointer field* igual a 183, razón por la cual se da un tratamiento especial a la finalización de la sección en este caso.

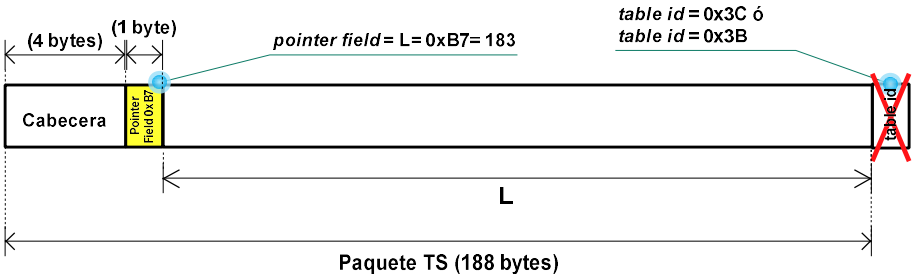


Figura 12. Uso incorrecto del *pointer field* en caso 1

Como se observa en la figura 12, el valor de 183 del *pointer field* cumple con la condición de indicar el número de bytes de la parte final de la sección pero por otro lado la ubicación del *pointer field* excede la capacidad el paquete (188 bytes) por lo que no es posible utilizar un *pointer field* con tal valor. En este caso se divide el final e inicio de la siguiente sección en paquetes diferentes, ubicando un byte de relleno (0xFF) y *pointer field* en dichos paquetes respectivamente. La figura 13 muestra el procedimiento de división descrito anteriormente.

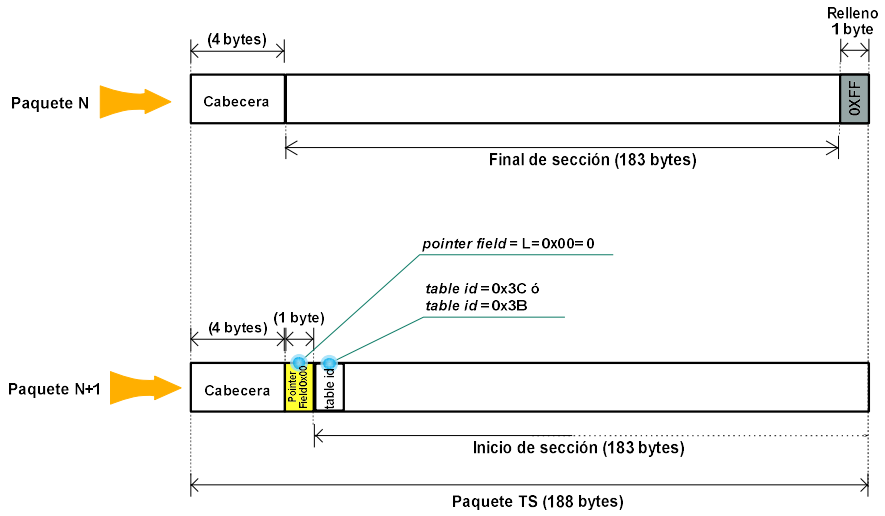


Figura 13. División de paquetes en caso 1

La figura 13 muestra el paquete N que contiene el final de una sección (183 bytes) más un byte de relleno para poder completar los 188 bytes del paquete, mientras que en el paquete N+1 será el que contenga el inicio de la siguiente sección marcado con un *pointer field* de cero (0). Es decir que el *pointer field* se verá obligado a desplazarse un paquete más adelante de lo que se podría haber calculado con el método mostrado mediante la ecuación 1. En *celinaginga.ts* se repiten varias veces el caso descrito anteriormente de los cuales uno se muestra en la figura 14, en donde el final de la sección 4 (del módulo 7) presenta un relleno al final del paquete (N) que lo contiene, mientras que el inicio de la sección 5 tiene que desplazarse al siguiente paquete (N+1) con un *pointer field* de cero.

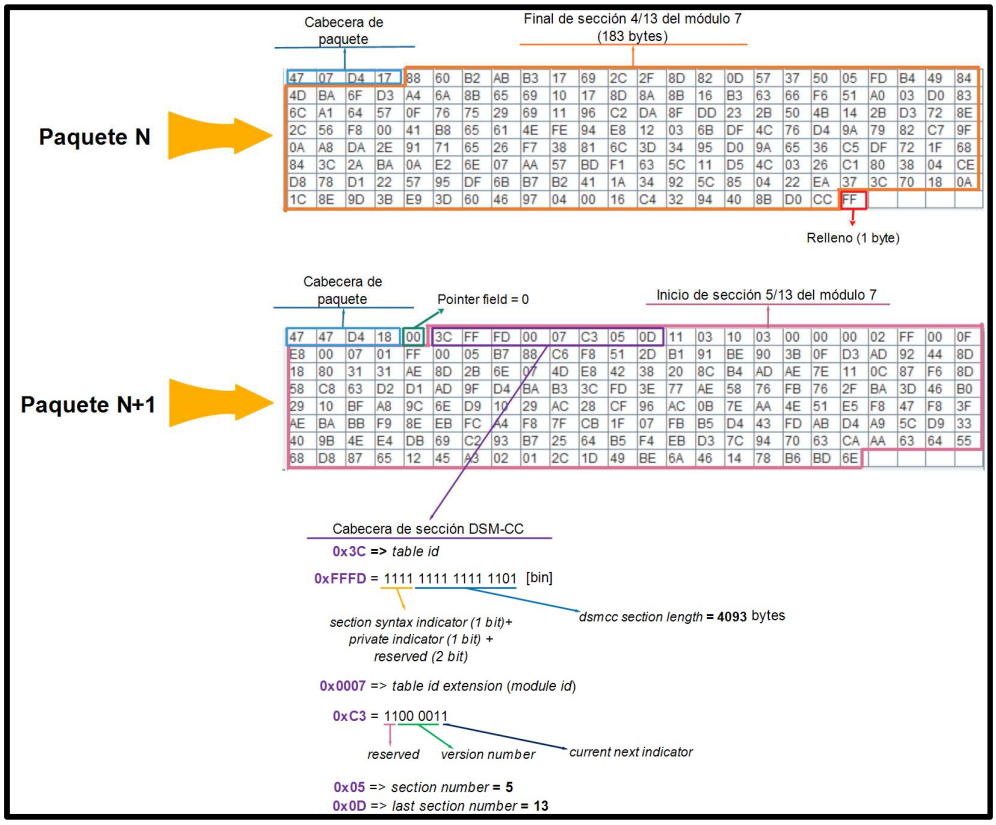


Figura 14. Caso 1 en módulo 7 de *celinaginga.ts*

3.1.3.2 Caso 2: Único *table id* de una sección en un paquete TS

Un paquete del TS solo puede poseer un único *pointer field* lo que obliga a que solo pueda existir un único *table id* que marque el inicio de una nueva sección dentro del mismo paquete. Sin embargo hay casos en los que la última sección que conforma un módulo es tan pequeña que esta inicia y termina en el mismo paquete, en cuyo caso se considera el uso de un relleno para completar el paquete e iniciar en otro paquete con *pointer field* de cero la siguiente sección. La figura 15 muestra la explicación del uso del relleno en un paquete donde se presenta este caso. El cálculo de relleno se basa principalmente en el uso del valor del *pointer field* (L) y el tamaño total de la sección (m=*dsmcc section length*+3).

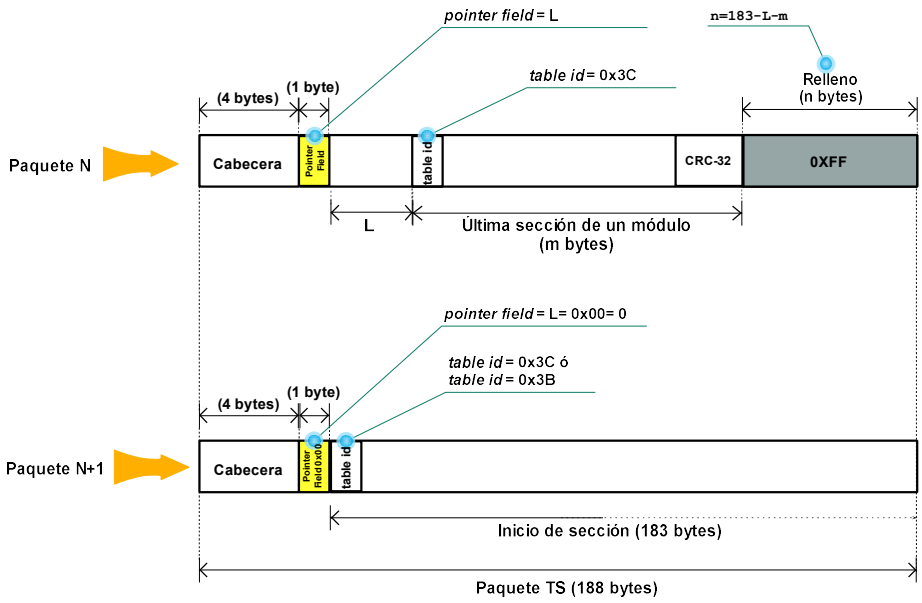


Figura 15. Uso y cálculo de relleno en el caso 2

La importancia de comprender tanto el caso 1 como el caso 2 radica en que para desencapsular las secciones no solo se debe analizar e eliminar cabeceras de cada sección, sino que también se debe tener la capacidad de descartar aquellos bytes de relleno los cuales no son considerados por ningún campo de la cabecera de los bloques que componen el carrusel de datos (DDB y DII), por lo que si estos bytes de relleno no fueran retirados en esta etapa, se generaría problemas o errores en la etapa de desencapsulación del carrusel de datos.

3.1.4. Algoritmo extractor de secciones DSM-CC

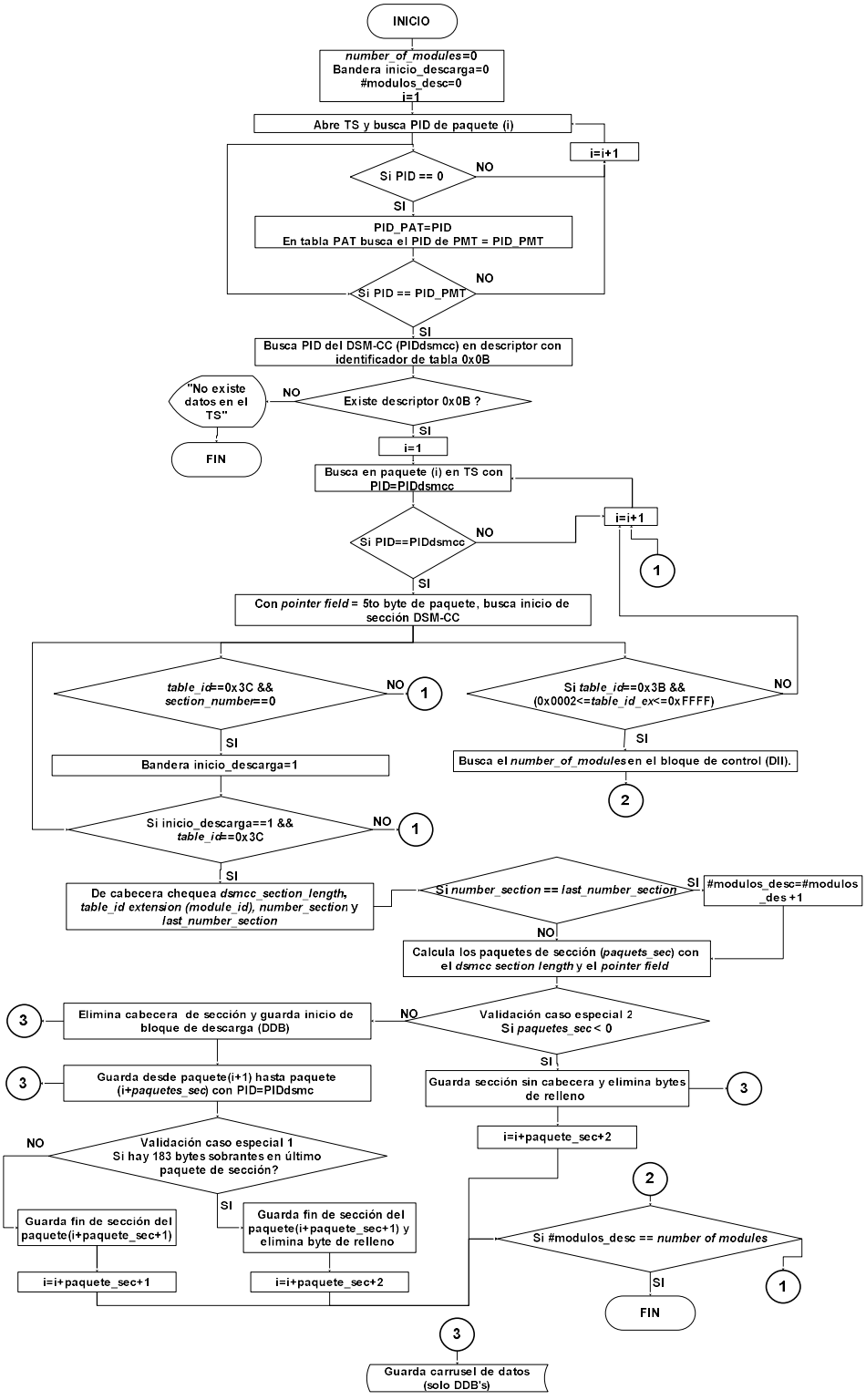


Figura 16. Diagrama de flujo de algoritmo extractor de secciones DSM-CC

Una vez analizada la estructura de las cabeceras de sección (control y descarga), generando además una forma de cálculo del número de paquetes entre el inicio y final de una sección, así como el tomar en consideración los casos especiales descritos anteriormente se desarrolló el algoritmo mostrado en la figura 16 el cual constituye la etapa 1 del proceso de extracción.

3.1.5. Desencapsulación del carrusel de datos

Una vez realizada la desencapsulación de secciones DSM-CC se obtiene el carrusel de datos formado únicamente por bloques DDB's. La desencapsulación del carrusel de datos (etapa 2 de proceso de extracción) se basa por lo tanto en el análisis y eliminación de las cabeceras de los bloques que componen cada módulo. La figura 17 muestra el fichero que contiene el carrusel de datos de *celinaginga.ts* abierto con la herramienta de software *notepad*++ donde se puede visualizar en formato ASCII (por defecto) los bytes que componen la cabecera de un bloque de descarga.

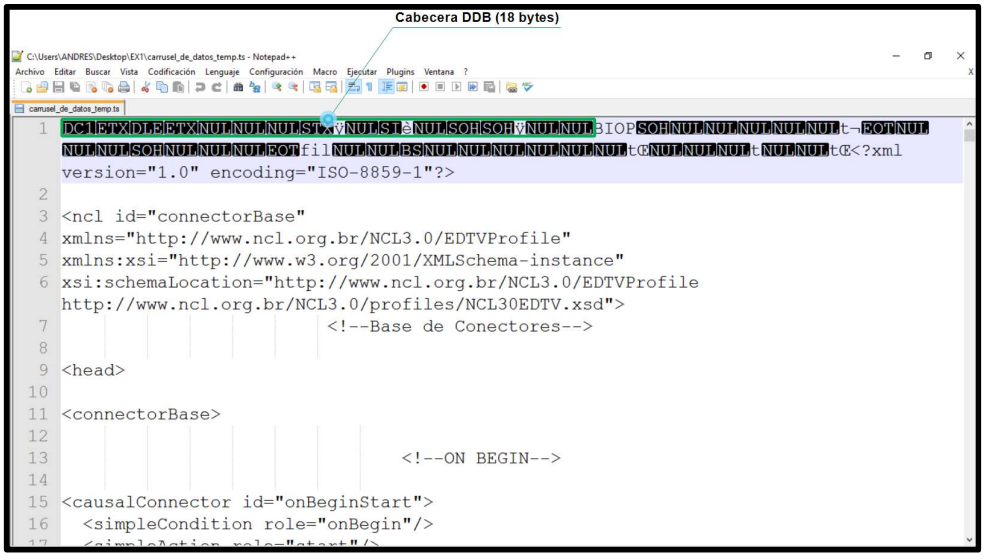


Figura 17. Visualización de cabecera DDB en *notepad*++

La figura 17 muestra la cabecera del primer DDB del carrusel de datos aunque adicionalmente también parte del contenido de carga útil de este bloque por lo que de alguna manera se puede prever que módulo al que pertenece dicho bloque, es el

encargado de transportar o contener el código fuente de la aplicación de GINGA-NCL. Para facilitar la visualización del análisis de las cabeceras se utilizó el *plugin* conversor de ASCII a hexadecimal con el que cuenta el software *notepad*++, por lo que de esta manera en la figura 18 se muestra el análisis de la cabecera en notación hexadecimal, haciendo énfasis en los campos que se consideró de uso necesario en el algoritmo extractor. El campo de uso principal en la desencapsulación de los bloques de descarga es el *Message Length* ya que permite encontrar el inicio de cada bloque, mientras que los campos *Module ID* y *Block Number* permiten mantener una referencia para dar seguimiento al estado de desencapsulación y saber cuándo debe finalizar la misma.

Figura 18. Análisis de cabecera DDB

3.1.6. Algoritmo extractor del carrusel de datos

La desencapsulación del carrusel de datos (etapa 2 del proceso de extracción) resulta muy sencilla siempre y cuando se haya eliminado o descartado correctamente los bytes de relleno que pudieron haber existido en los paquetes que contenían las secciones DSM-CC. Por lo que se elaboró un algoritmo que se basa principalmente en el uso del tamaño del bloque (*Message Length*) y el identificador del módulo (*Module*

Id) para cumplir con la etapa 2 de extracción de datos. La figura 19 muestra el algoritmo de extracción del carrusel de datos que como se observa también considera un valor encontrado durante la desencapsulación de las secciones (etapa 1) como lo es el número de total de módulos (*number of modules)* y donde se obtiene como resultado final un fichero que contiene únicamente el carrusel de objetos.

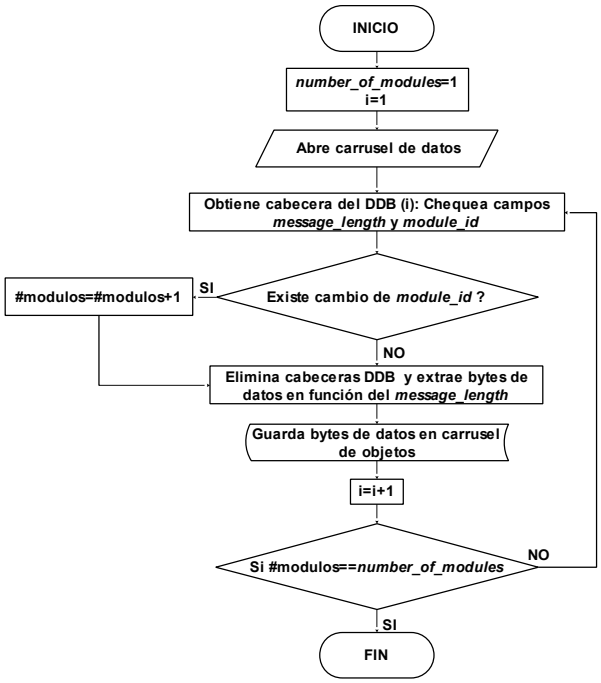


Figura 19. Diagrama de flujo de algoritmo extractor del carrusel de datos

3.1.7. Desencapsulación del carrusel de objetos

La desencapsulación del carrusel de objetos (etapa 3 del proceso de extracción) tiene como objetivo principal recuperar el directorio completo de la aplicación así como también su contenido (código fuente, archivos de texto o multimedia, etc), para lo cual fue necesario dividir la etapa 3 de extracción en 3 subprocesos bien definidos:

- IAA (Identificación, Análisis y Almacenamiento temporal) de objetos tipo fichero (*file*).
- IAA de objetos de tipo directorio (*directory*) y puerta de servicio (*service gateway*).
- Relacionamiento de CO (Claves de Objeto) de directorio y componentes asociados para construcción del directorio.

3.1.7.1 IAA de objetos tipo fichero

Los objetos BIOP tipo fichero (‘fil’) se encargan principalmente de transportar el contenido de un fichero independientemente del tipo de fichero o formato que este posea, por lo que de la cabecera de este tipo de objetos lo más importante (para el algoritmo extractor) es localizar su clave de objeto (*Object key data byte*) debido a que este parámetro es el único que permite relacionar el contenido del fichero con su nombre y además con la carpeta a la que pertenece. También en necesario utilizar el campo *Message Size* para saber cuántos bytes son los que conforman el objeto (fichero y cabecera). La figura 20 muestra el análisis que se debe hacer (de acuerdo a la estructura de un objeto tipo *file*) para localizar los dos parámetros antes mencionados utilizando el primero objeto del carrusel de *celinaginga.ts*.

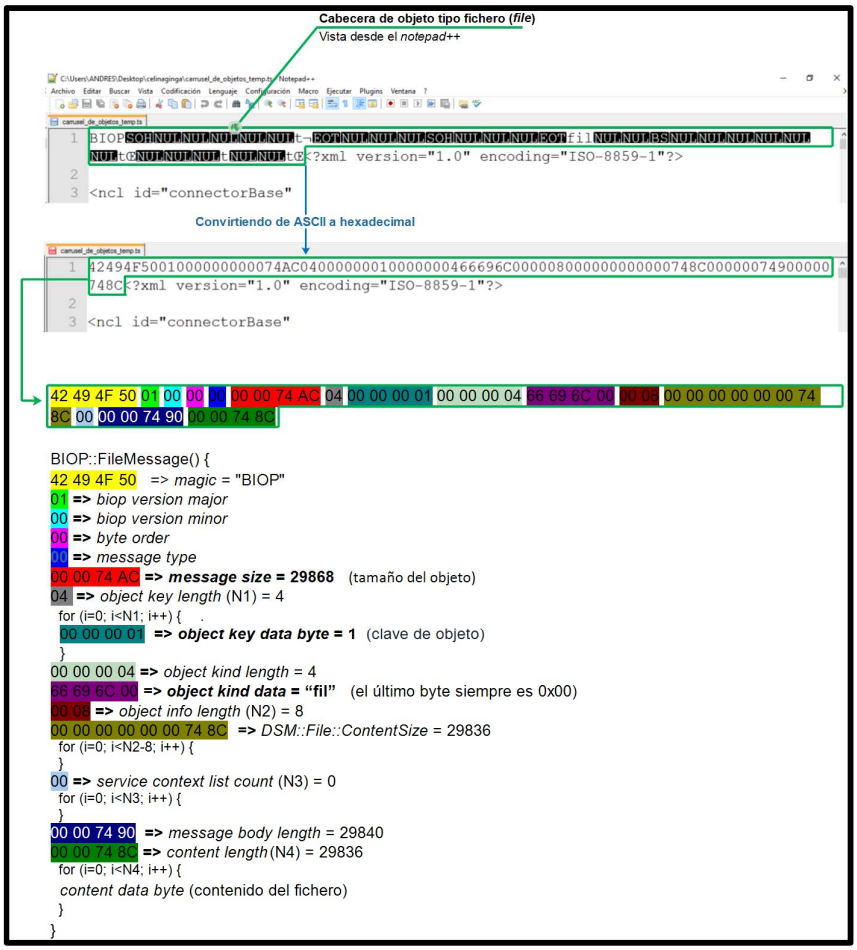


Figura 20. Análisis de cabecera BIOP tipo fichero

En el inicio del objeto fichero analizado se visualiza la palabra “mágica” BIOP para lo cual es importante mencionar que todo objeto (sea fichero, directorio o puerta de servicio) debe iniciar con esta palabra. La versión mayor y menor del objeto BIOP establece la versión principal y secundaria del mismo, por lo cual son valores que pueden ir actualizándose según continúe la transmisión del carrusel, sin embargo estos campos son puestos usualmente con 0x01 y 0x00 respectivamente, para la especificación de MPEG 2 TS mientras que el *byte order* está en 0x00 indicando que los bytes que siguen (en la cabecera) a partir de este campo se debe leer o interpretar considerando la codificación *big endian*, es decir que el bit o byte menos significativo se toma desde el lado derecho del campo que se esté analizando.

En caso de que el *byte order* se encontrara puesto en 0x01 entonces se debería considerar la codificación *little endian* (bit o byte menos significativo a la izquierda), sin embargo para la especificación de MPEG-2 se usa siempre *big endian*, por lo que es posible asumir de antemano que los campos de la cabecera de objeto (incluyendo el *message size* y el *object key data byte*) se deben entender de acuerdo a dicha codificación.

Después del campo *message type* (siempre puesto en 0x00) se encuentra el *message size* con valor en este caso de 29868 bytes y el cual representa la longitud o número de bytes del objeto, los cuales son contabilizados a partir del siguiente campo (*object key length*) hasta el final del contenido del fichero (*content data byte*). Esta longitud es la principal usada en el algoritmo de desencapsulación del carrusel objetos ya que esta se ubica siempre en la misma posición de la cabecera de objeto independientemente del tipo que sea y a su vez concede una visión global del tamaño del objeto, no así el campo *message body length* cuya posición depende particularmente del tipo de objeto.

El *object kind data* al ser un objeto tipo fichero contiene la cadena de bytes 0x66696C00 (“fil” en ASCII) lo cual proporciona la forma de clasificar los objetos para un procesamiento adecuado de la información que brindan sus cabeceras a partir

de este campo, considerando que la estructura del objeto fichero desde este punto cambia mucho en comparación con el de directorio.

El *DSM File ContentSize, content length* y *message body length* (menos 4 bytes) describen el número de bytes que pertenecen únicamente al contenido del fichero (*content data file*) que el caso analizado es de 29836 bytes, sin embargo no es necesario utilizarlos en el algoritmo extractor (para objetos tipo *file*) ya que el mismo número de bytes puede ser calculado con el campo *message size* al restarle los bytes de la cabecera que le siguen a este campo (32 bytes). En cuanto al campo *service context list count* describe al número de bytes de datos privados del usuario pero para la especificación de MPEG2 TS este campo siempre será puesto en cero.

El almacenamiento temporal del contenido y clave del objeto se basa principalmente en guardar todos los objetos tipo fichero encontrados en vectores o matrices "objeto" en donde la primera posición del vector estará reservada para la clave de objeto y a continuación se ubicará el contenido del fichero, con el objetivo de que al finalizar el proceso de clasificación de objetos se facilite el relacionamiento de claves de objeto con su respectivo nombre y directorio. La figura 21 muestra la conformación de un vector "objeto" considerando los valores clave de objeto y contenido hallados en el análisis de la figura 20.

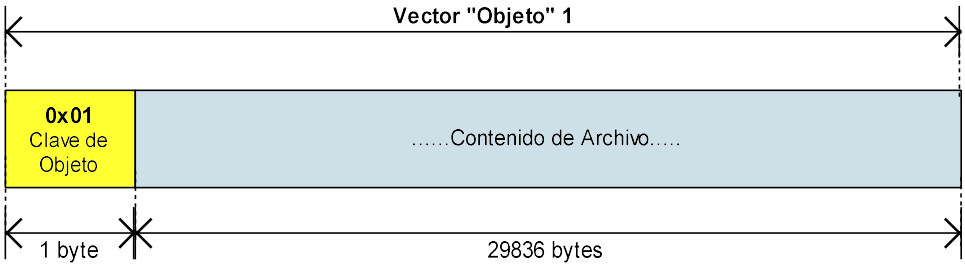


Figura 21. Conformación de vector "objeto"

3.1.7.2 IAA de objetos de tipo directorio y puerta de servicio

Los objetos BIOP tipo directorio o puerta de enlace mantienen una idéntica estructura y se los puede identificar, analizar y almacenar de la misma manera por lo que durante la explicación de este proceso se referirá a ambos tipos de objeto

únicamente como tipo directorio. Este tipo de objeto también tiene la característica de tener un tamaño muy inferior a los de tipo fichero ya que se encargan de transportar únicamente el nombre de los ficheros y directorios además de sus respectivas claves de objeto. La figura 22 muestra el análisis de un objeto directorio de *celinaginga.ts*.

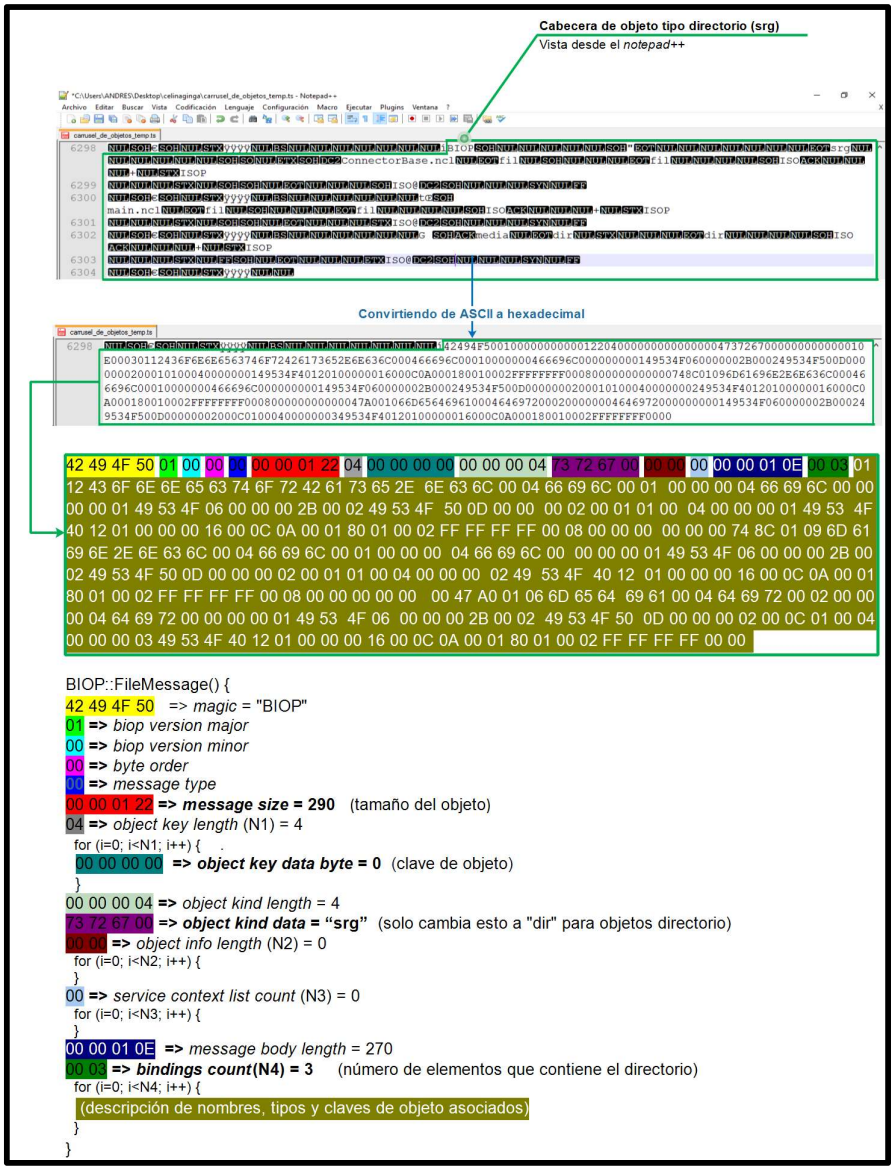


Figura 22. Análisis de cabecera de objeto tipo directorio

En la figura 22 se realiza un análisis idéntico para encontrar el tamaño, clave y tipo de objeto considerando que este último parámetro puede ser "*dir*" o "*srg*" según sea directorio o puerta de servicio, siendo esto la única diferencia entre las estructuras de

ambos tipos de objeto. El número de elementos que contiene el directorio lo da el campo *bindings count* (N4) que en este caso será igual a 3, lo que nos indica que el dicho directorio contiene 3 componentes entre archivos y carpetas. La búsqueda de nombre, tipo y clave de objeto asociado de los componentes 1, 2 y 3 es mostrada en las figuras 23, 24 y 25 respectivamente.

Figura 23. Análisis de componente 1

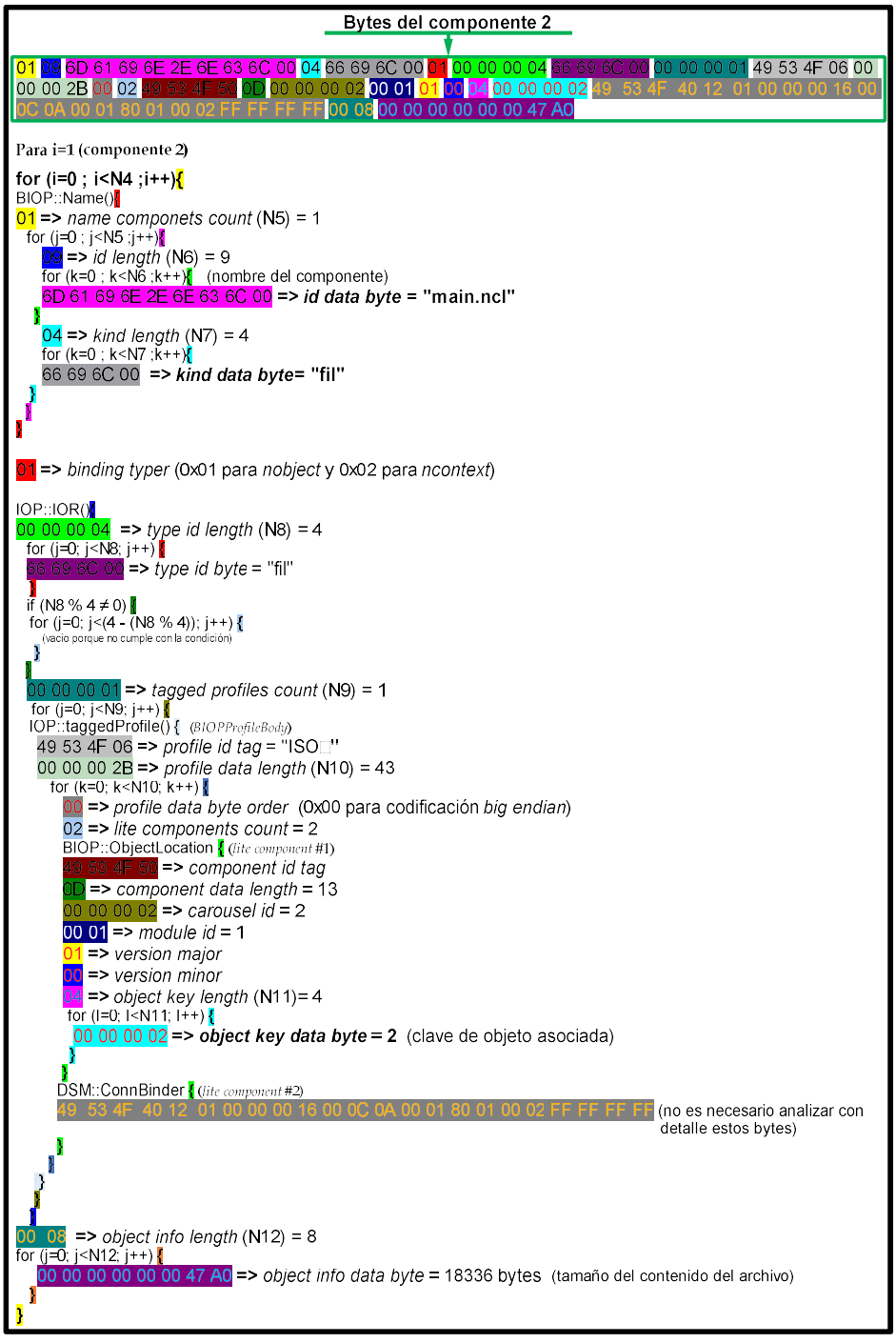


Figura 24. Análisis de componente 2

Figura 25. Análisis de componente 3

Para el algoritmo planteado es necesario identificar el tipo de objeto de todos los componentes del directorio con cualquiera de los campos *kind data byte*, *type id byte* o *binding typer* con valores de 0x01 (*nobject*) para fichero y 0x02 (*ncontext*) para

directorio en este último campo. Después se debe almacenar temporalmente los datos recopilados de clave de objeto del directorio, clave de objeto de componentes asociados y su respectivo nombre (*id data byte*) en vectores “nombre fichero” y “nombre carpeta” según el tipo de componente. La figura 26 muestra la estructura que se propone para dichos vectores utilizando como referencia los datos obtenidos tras haber analizado el objeto directorio de la figura 22 y sus 3 componentes.

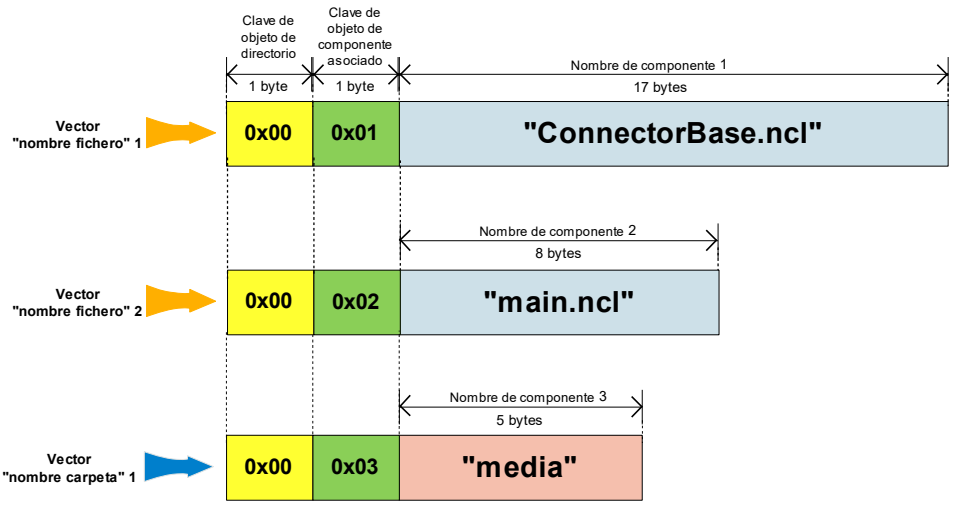


Figura 26. Conformación de vectores “nombre fichero” y “nombre carpeta”

Como se observa la estructura del vector “nombre fichero” y “nombre carpeta” es semejante a la del vector “objeto”, con la diferencia de que en este caso se debe almacenar dos claves de objeto (de directorio y componente) en los dos primeros bytes del vector. La forma de relacionar estos vectores para construir el directorio de la aplicación se puede observar entre el vector “objeto” 1 (figura 21) y el vector “fichero” 1 (figura 26) donde al coincidir la clave de objeto del fichero (0x01) con la clave de objeto del componente (0x01) se sabe que el contenido del fichero le corresponde el nombre “ConnectorBase.ncl” y al analizar el clave del directorio (0x00) se sabe que este fichero se debe ubicar en el directorio raíz de la aplicación. Una explicación más detallada de la forma propuesta de construcción total del directorio (después de identificar, analizar y almacenar en vectores todos los objetos del carrusel) se muestra a continuación.

3.1.7.3 Relacionamiento de CO de directorio y componentes asociados

El proceso de construcción del directorio completo de la aplicación relacionando claves de objeto a partir de los vectores “objeto”, “nombre fichero” y “nombre carpeta” consta de los siguientes pasos.

- Identificar las carpetas finales (carpetas que no contienen otras carpetas).
- Encontrar rutas (*path*) de todas las carpetas finales en relación al directorio raíz.
- Crear directorios en función de las rutas encontradas.
- Relacionar ficheros con su nombre y encontrar su ubicación en relación a los directorios creados.
- Colocar los ficheros con su nombre dentro de su respectivo directorio.

Se explicará a continuación dichos pasos suponiendo el ejemplo de la figura 27 donde tenemos un conjunto de vectores “objeto”, “nombre fichero” y “nombre carpeta” que representa a un directorio completo de una aplicación GINGA-NCL.

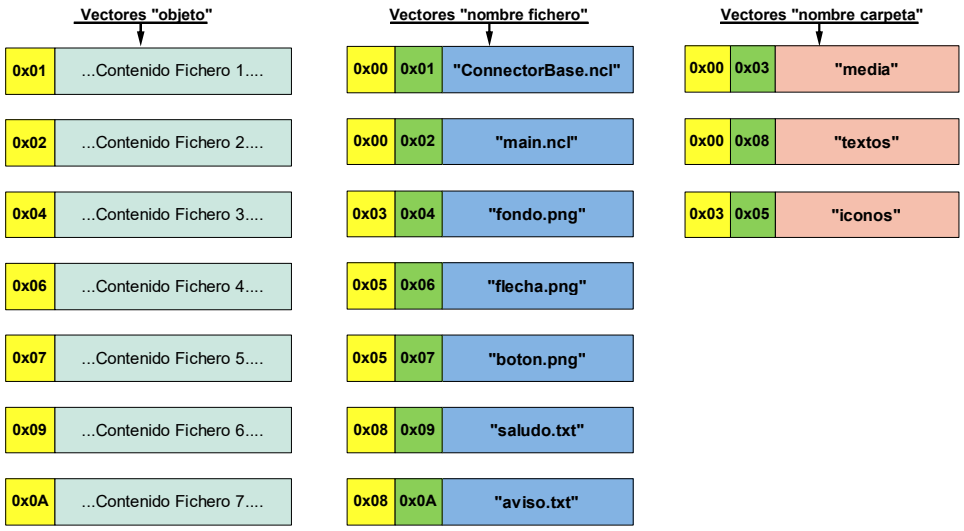


Figura 27. Vectores para explicación de método de creación de directorio

Crear el directorio completo requerirá de generar ciertas rutas o caminos de una carpeta hasta la raíz, sin embargo para optimizar y reducir la cantidad de rutas a lo mínimo necesario se requiere identificar las carpetas finales, es decir aquellas carpetas que no contienen otras carpetas dentro de sí. Estas carpetas finales se pueden identificar de dos formas: La primera durante la etapa de análisis de objetos tipo

directorio, reconociendo a una carpeta final como aquel directorio que no puede poseer ningún componente asociado tipo directorio y la segunda mediante el análisis de los vectores “nombre carpeta”.

La forma de reconocer una carpeta final a través de los vectores “nombre carpeta” es descartando primero a aquellas carpetas que no lo son. Las carpetas que no son finales son todas aquellas cuya clave de objeto asociado coincide con alguna de las claves de objeto del directorio, lo que deja a las que no cumplan con esto, como carpetas finales. La figura 28 muestra esta forma de identificación de carpetas finales, a través del ejemplo propuesto.

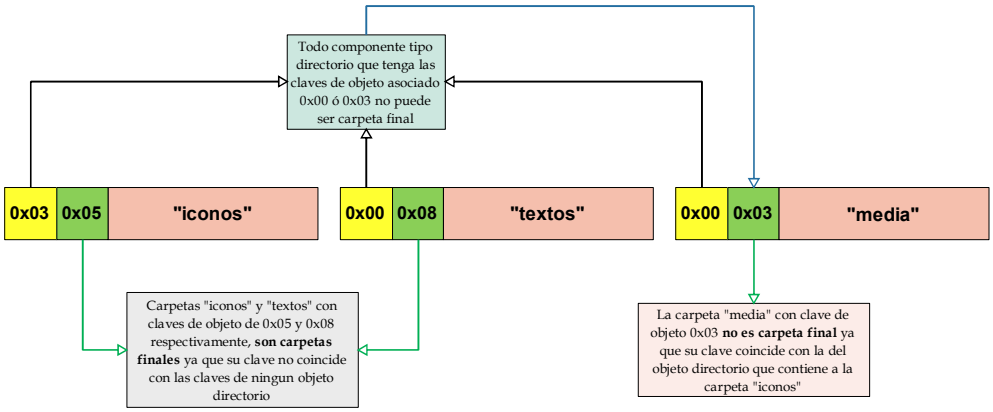


Figura 28. Análisis para identificación de carpetas finales

De análisis hecho en la figura 28, se puede sacar 3 conclusiones importantes: La primera es que las carpetas finales corresponden a las carpetas “íconos” y “textos” con claves de objeto asociado 0x05 y 0x08 respectivamente, la segunda es que la carpeta “media” no es final dado que al coincidir su clave de objeto asociado con la de directorio de la carpeta “íconos”, da a entender que esta última está contenida dentro de la carpeta “media” y la tercera es que si bien es cierto no hay ninguna clave de objeto asociado que coincida con la clave de directorio 0x00 no quiere decir que esta corresponda a una carpeta final, sino mas bien significa que la clave 0x00 le corresponde al directorio raíz el cual no posee un nombre determinado como las demás carpetas, de hecho la única forma para que el directorio raíz sea considerado como carpeta final es que no se haya creado ningún vector “nombre carpeta” después de haber analizado e identificado todos los objetos del carrusel.

Después de haber identificado las carpetas finales y sus respectivas claves de objeto se deben encontrar rutas desde estas carpetas finales hacia el directorio raíz el cual de antemano sabremos que su clave de objeto siempre es cero (0x00). La figura 29 muestra la búsqueda de la ruta 1 para la carpeta final “íconos”, la cual se realiza relacionando claves de objeto de directorio y componentes asociados.

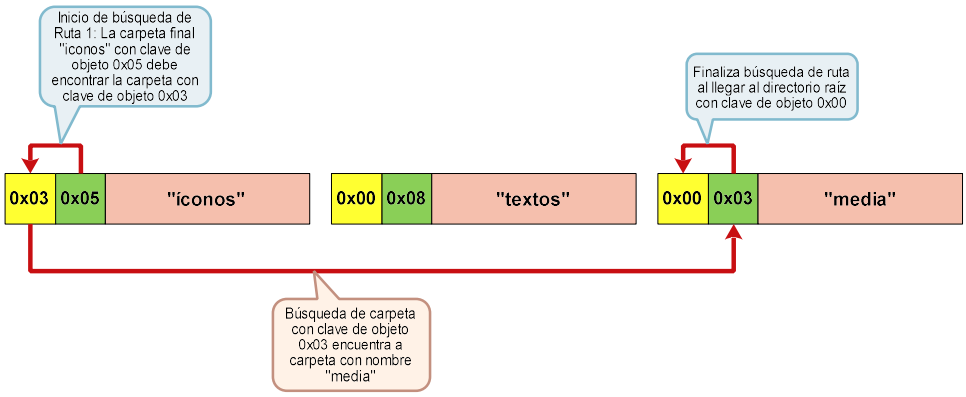


Figura 29. Búsqueda de ruta 1 para carpeta “íconos”

En general la búsqueda de rutas independientemente de la complejidad del directorio de la aplicación, se realiza de la misma forma en todos los casos ya sea que se encuentre una ruta extensa que involucre muchas claves de objeto o pocas como la que se muestra en la figura 30 al realizar la búsqueda de la ruta 2 para la carpeta final “textos”, donde únicamente relaciona su clave de objeto 0x08 con la de la raíz (0x00) terminando rápidamente con la búsqueda de ruta.

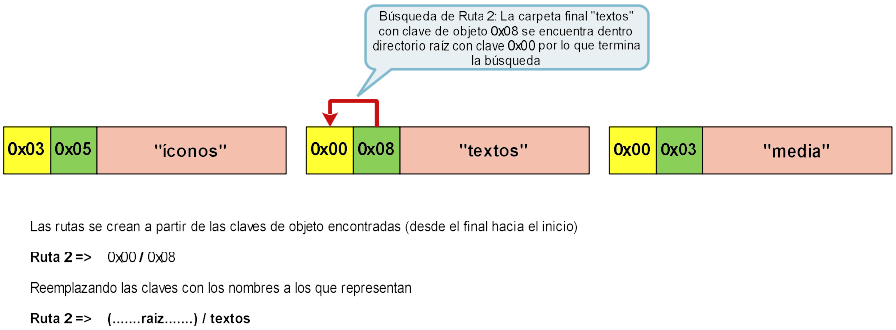


Figura 30. Búsqueda de ruta 2 para carpeta "textos"

Independientemente del lenguaje de programación (Matlab, java, Python, C++, etc) o del tipo de sistema operativo (Windows, Linux, Mac OS, etc) con el que se requiera crear un directorio, se debe considerar el método de creación de directorios de forma recursiva, la cual se caracteriza por poder crear cadenas de directorios anidados independientemente si existan o no anteriormente estos directorios. De esta manera la creación de un directorio a partir de rutas encontradas con carpetas finales, resulta muy sencilla. La figura 31 muestra el resultado final de creación del directorio de la aplicación a partir de la Ruta 1 y Ruta 2, considerando el uso de sentencias en Matlab.

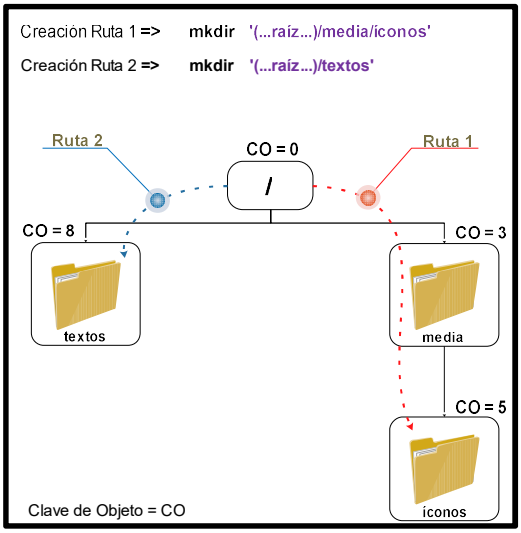


Figura 31. Creación de directorio con rutas encontradas

Una vez creado el directorio, se puede ubicar los ficheros dentro de su respectiva carpeta, pero para ello primero es necesario asociar el contenido de cada fichero

(vector “objeto”) con su nombre correspondiente (vector “nombre fichero”) relacionándolos mediante sus claves de objeto. Después se debe generar una ruta de su ubicación de manera similar a cuando se encontró las rutas para crear el directorio. La figura 32 muestra cómo se relacionan los vectores para cumplir con este objetivo.

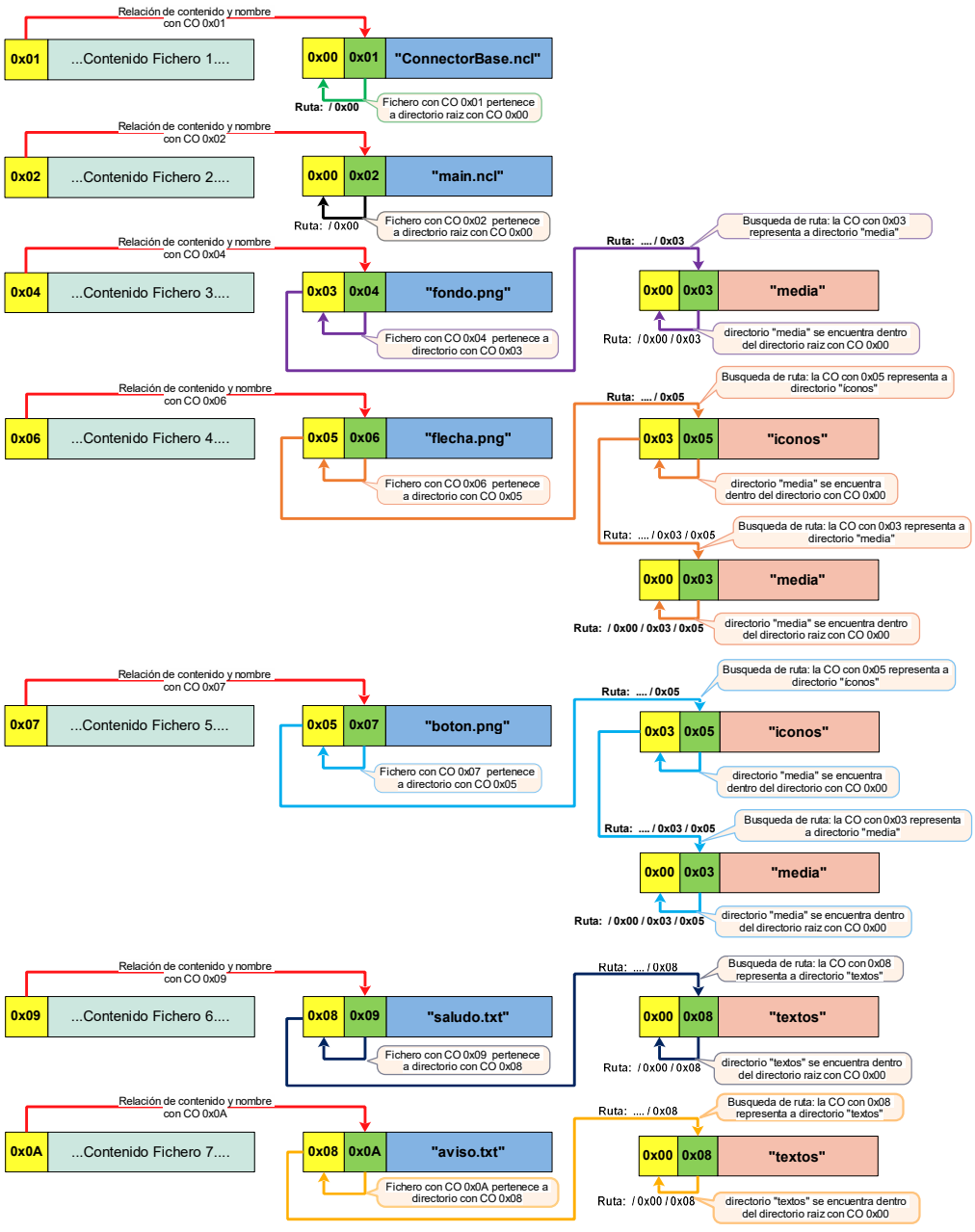


Figura 32. Creación de rutas para ficheros

Las rutas creadas para cada fichero por lo tanto son las siguientes:

- ConnectorBase.ncl → Ruta: /0x00 = /(…raíz…)
- main.ncl → Ruta: /0x00 = /(…raíz…)
- fondo.png → Ruta: /0x00/0x03 = /(…raíz…)/media
- flecha.png → Ruta: /0x00/0x03/0x05 = /(…raíz…)/media/íconos
- boton.png → Ruta: /0x00/0x03/0x05 = /(…raíz…)/media/íconos
- saludo.txt → Ruta: /0x00/0x08 = /(…raíz…)/textos
- aviso.txt → Ruta: /0x00/0x08 = /(…raíz…)/textos

De acuerdo a la ruta cada fichero, este guardado en su directorio como se muestra en la figura 33, donde se visualiza el directorio de la aplicación construida en su totalidad.

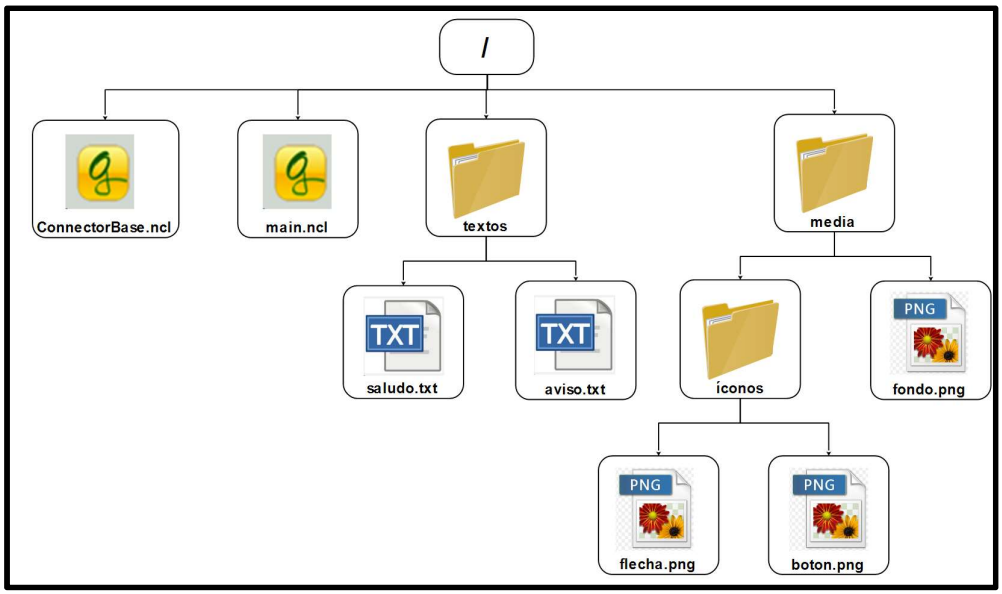


Figura 33. Directorio construido de la aplicación

3.1.8. Algoritmo extractor del carrusel de objetos

La figura 34 muestra el algoritmo de extracción del carrusel de objetos, el cual se basa en el procesamiento del carrusel de objetos que se obtuvo como resultado de la aplicación de algoritmo extractor del carrusel de datos (figura 19) y que cumple con los 3 subprocesos de extracción propuestos anteriormente.

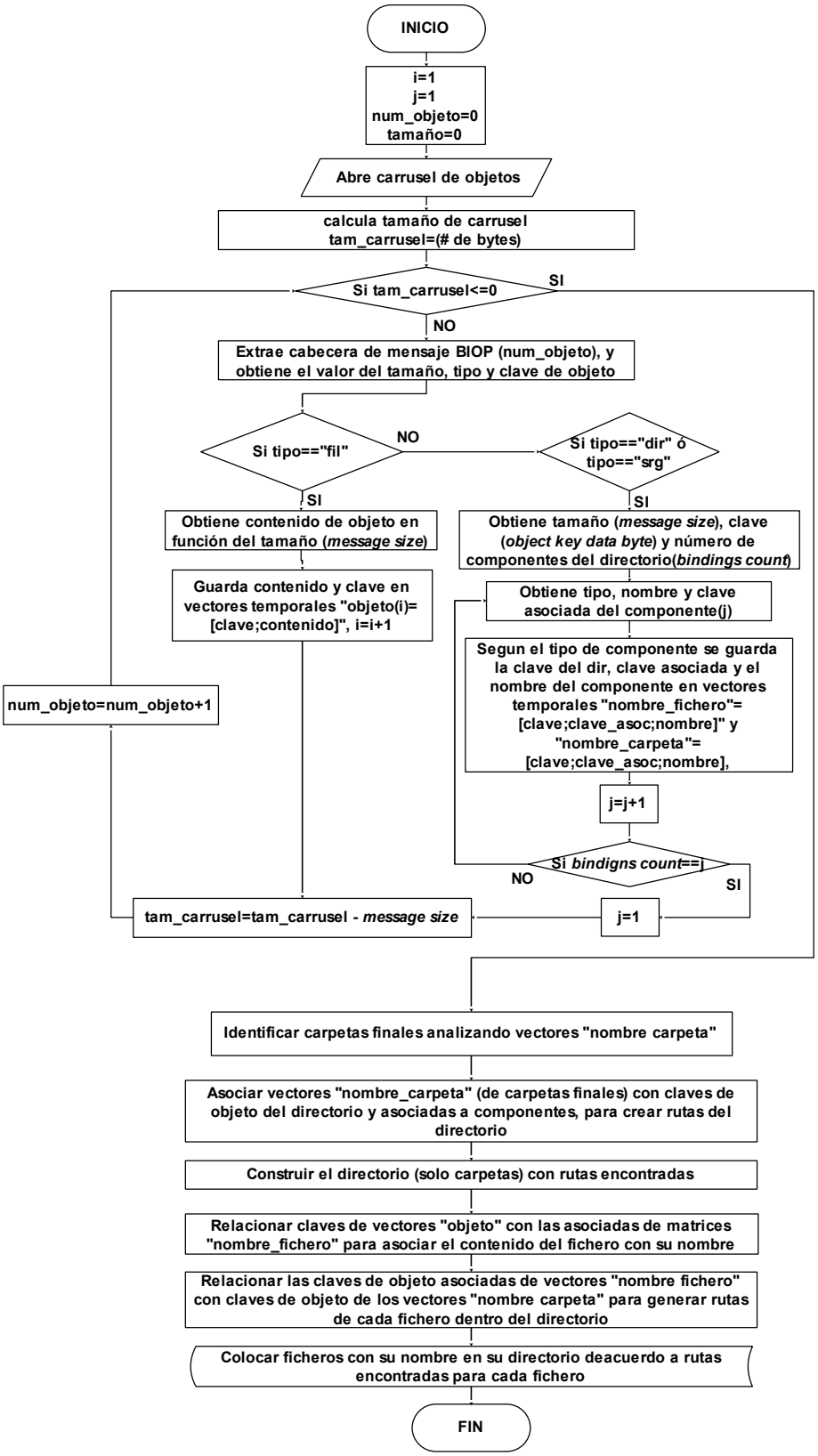


Figura 34. Diagrama de flujo de algoritmo extractor del carrusel de objetos

3.2. Elaboración del software en MATLAB

3.2.1. Funciones en MATLAB

Para elaborar el software extractor y constructor de datos se implementó cada etapa del algoritmo desarrollado (extracción de secciones DSM-CC, extracción del carrusel de datos y extracción del carrusel de objetos) en 3 funciones principales de MATLAB (Haubold, 2007), una para cada etapa y además otras funciones secundarias que sirven para lectura de campos de cabecera de sección, bloque DDB y objeto. La figura 35 muestra las funciones implementadas.

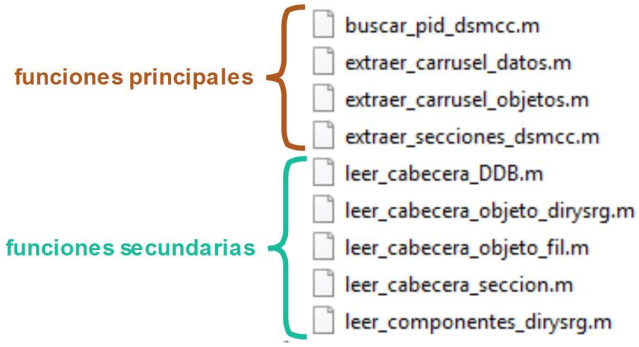


Figura 35. Funciones de en MATLAB del software extractor

Cabe mencionar que la función *buscar_pid_dsmcc.m* se la consideró como principal ya que se encuentra dentro del proceso de extracción de secciones DSM-CC y se caracteriza por que cumple con la búsqueda del PID correspondiente a los paquetes del DSM-CC. También se implementó una función secundaria *leer_componentes_dirysrg.m* que cumple por otro lado con la lectura de campos que describen a los componentes contenidos en los objetos tipo directorio y puerta de servicio.

3.2.2. Interfaz Gráfica de Usuario (GUI)

Para facilitar el uso de las funciones implementadas se diseñó un entorno gráfico de usuario para el software el cual se muestra en la figura 36. El entorno gráfico

muestra opciones como el botón “Abrir” (1) que sirve para escoger el TS del cual se va a extraer la información y cuyo directorio se visualizará en el área de texto de la parte superior de (2). El botón “Salida” (3) permite seleccionar el directorio destino (raíz) de la extracción y en donde además se guardarán tanto fichero temporales del carrusel de datos y objetos, como un reporte detallado del proceso de extracción presentado en una hoja de cálculo de Excel(*.xlsx), en caso de que se requiera. El directorio destino se podrá visualizar en el área de texto de la parte inferior de (2). El botón “Extraer” (4) inicia el proceso de extracción, mientras que en (5) se presenta algunas opciones adicionales tales como si se quiere o no eliminar archivos temporales, generar archivo de reporte detallado o abrir la carpeta de extracción al finalizar el proceso.

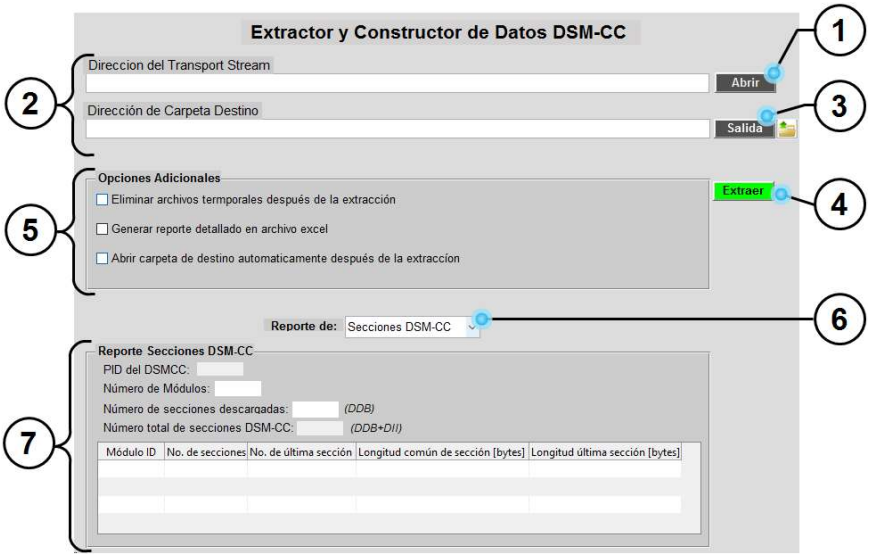


Figura 36. GUI del software extractor

El menú de opciones (7) permite seleccionar el tipo de reporte resumen que se desea visualizar (Secciones DSM-CC, carrusel de datos o carrusel de objetos). Por defecto se encuentra seleccionado el reporte de la Secciones DSM-CC, el cual muestra datos como PID de los paquetes DSM-CC, número de módulos extraídos, número de mensajes de descarga y control encontrados. También muestra una tabla con datos referentes los módulos extraídos.

3.2.3. Funcionamiento del software: GUI, reporte resumido y detallado

El objetivo principal del software es el de extraer y construir los datos de las aplicaciones que se transmiten en un TS, sin embargo también se implementó en software la función de generar reportes resumidos en el GUI y reportes detallados a través de un archivo de Excel los cuales servirán para constatar y analizar el proceso de extracción de los datos según la teoría expuesta.

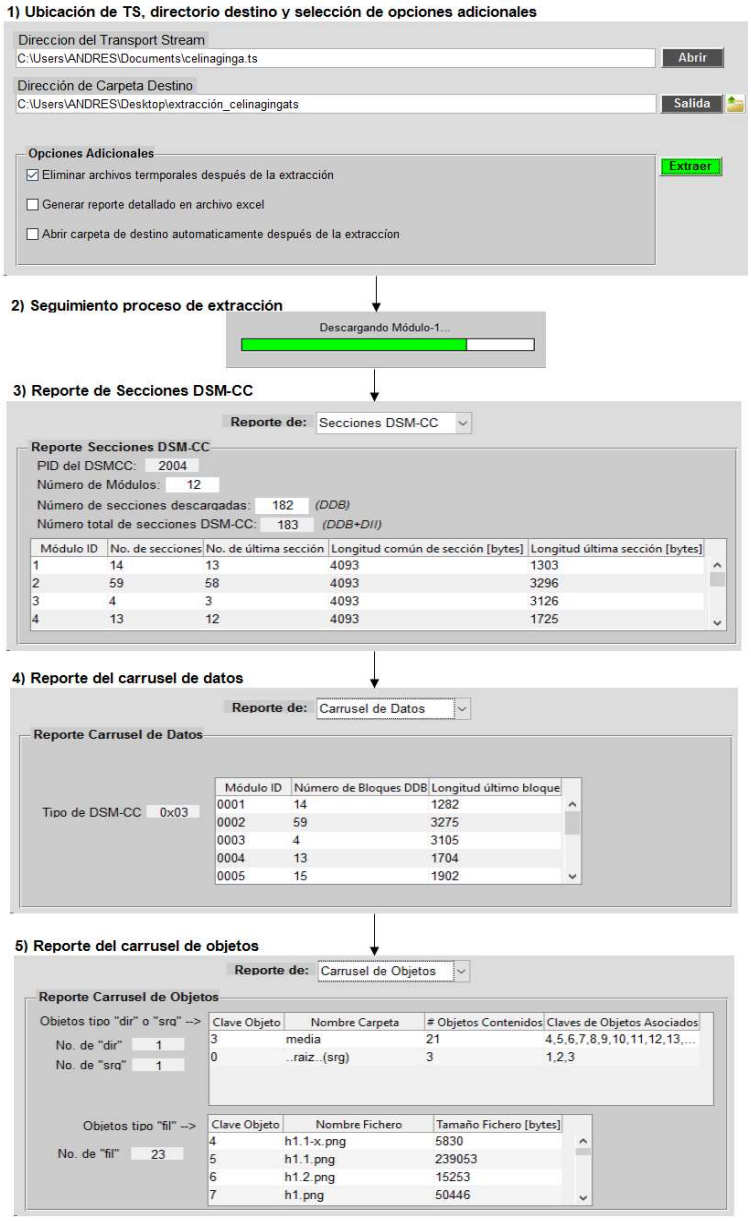
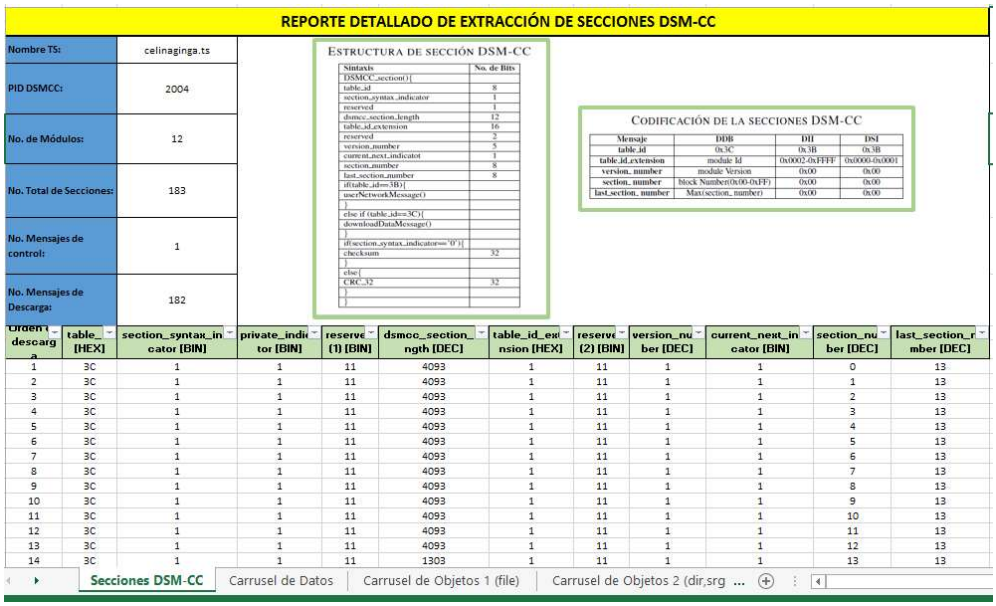


Figura 37. Resultado de implementación del *software* extractor

La figura 37 muestra los resultados de la implementación del software, con el TS de prueba *celinaginga.ts* y los reportes resumen que se obtuvieron al finalizar el proceso de extracción, de Secciones DSM-CC, carrusel de datos y carrusel de objetos.

El reporte resumido de carrusel de datos muestra el tipo de mensajes DSM-CC del carrusel (0x03 para Mensajes U-N DSM-CC), así como también información de los DDB's pertenecientes a cada módulo. Por otro lado el reporte resumido de carrusel de objetos muestra los mensajes "*srg*", "*dir*" y "*fil*" encontrados, además de sus claves de objeto propias y asociadas. Los resultados de estos reportes y extracción final de datos son analizados en el Capítulo 4.

El reporte detallado de la extracción presenta datos referentes a la descarga de cada sección, bloque u objeto y la lectura de cada campo de sus cabeceras, dando la posibilidad de realizar un análisis más profundo de la información en relación a cada etapa de desencapsulación de datos. La figura 38 muestra el formato del reporte detallado de la extracción de secciones DSM-CC donde en la parte superior izquierda se visualiza datos generales de la extracción mientras que en la parte superior derecha se muestra las tablas que contienen la estructura de las secciones y codificación de algunos campos de la misma, las cuales permiten rápidamente analizar y comparar con los datos recopilados de cada sección (parte inferior del reporte), donde cada fila representa a una sección descargada.



REPORTE DETALLADO DE EXTRACCIÓN DE SECCIONES DSM-CC

Nombre TS:	celinaginga.ts
PID DSMCC:	2004
No. de Módulos:	12
No. Total de Secciones:	183
No. Mensajes de control:	1
No. Mensajes de Descarga:	182

ESTRUCTURA DE SECCIÓN DSM-CC

Sintaxis	No. de Bits
DSMCC_section(){	
table_id	8
section_syntax_indicator	1
reserved	1
dsmcc_section_length	12
table_id_extension	16
reserved	2
version_number	5
current_next_indicator	1
section_number	8
last_section_number	8
if(table_id==3B){	
userNetworkMessage()	
}	
else if (table_id==3C){	
downloadDataMessage()	
}	
if(section_syntax_indicator=='0'){	
checksum	32
}	
else{	
CRC_32	32
}	
}	

CODIFICACIÓN DE LA SECCIONES DSM-CC

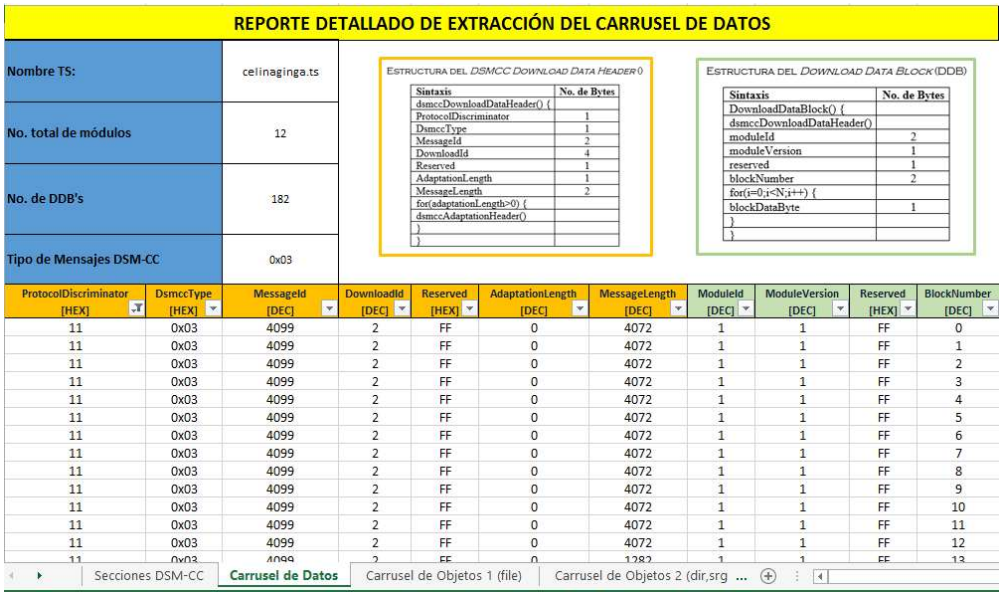
Mensaje	DDB	DII	DSI
table_id	0x3C	0x3B	0x3B
table_id_extension	module Id	0x0002-0xFFFF	0x0000-0x0001
version_number	module Version	0x00	0x00
section_number	block Number(0x00-0xFF)	0x00	0x00
last_section_number	Max(section_number)	0x00	0x00

Orden descarga	table_ [HEX]	section_syntax_indicator [BIN]	private_indicator [BIN]	reserved (1) [BIN]	dsmcc_section_length [DEC]	table_id_extension [HEX]	reserved (2) [BIN]	version_number [DEC]	current_next_indicator [BIN]	section_number [DEC]	last_section_number [DEC]
1	3C	1	1	11	4093	1	11	1	1	0	13
2	3C	1	1	11	4093	1	11	1	1	1	13
3	3C	1	1	11	4093	1	11	1	1	2	13
4	3C	1	1	11	4093	1	11	1	1	3	13
5	3C	1	1	11	4093	1	11	1	1	4	13
6	3C	1	1	11	4093	1	11	1	1	5	13
7	3C	1	1	11	4093	1	11	1	1	6	13
8	3C	1	1	11	4093	1	11	1	1	7	13
9	3C	1	1	11	4093	1	11	1	1	8	13
10	3C	1	1	11	4093	1	11	1	1	9	13
11	3C	1	1	11	4093	1	11	1	1	10	13
12	3C	1	1	11	4093	1	11	1	1	11	13
13	3C	1	1	11	4093	1	11	1	1	12	13
14	3C	1	1	11	1303	1	11	1	1	13	13

Secciones DSM-CC | Carrusel de Datos | Carrusel de Objetos 1 (file) | Carrusel de Objetos 2 (dir,srg ...

Figura 38. Reporte detallado de extracción de secciones DSM-CC

De manera similar es el formato de presentación del reporte detallado tanto de la extracción del carrusel de datos como del carrusel de objetos, tal y como se muestra en las figuras 39, 40 y 41. En relación al reporte de extracción del carrusel de objetos se decidió presentarlo en 2 hojas distintas, una para objetos tipo fichero y otra para objetos tipo directorio.



REPORTE DETALLADO DE EXTRACCIÓN DEL CARRUSEL DE DATOS

Nombre TS:	celinaginga.ts
No. total de módulos	12
No. de DDB's	182
Tipo de Mensajes DSM-CC	0x03

ESTRUCTURA DEL *DSMCC DOWNLOAD DATA HEADER()*

Sintaxis	No. de Bytes
dsmccDownloadDataHeader() {	
ProtocolDiscriminator	1
DsmccType	1
MessageId	2
DownloadId	4
Reserved	1
AdaptationLength	1
MessageLength	2
for(adaptationLength>0) {	
dsmccAdaptationHeader()	
}	
}	

ESTRUCTURA DEL *DOWNLOAD DATA BLOCK* (DDB)

Sintaxis	No. de Bytes
DownloadDataBlock() {	
dsmccDownloadDataHeader()	
moduleId	2
moduleVersion	1
reserved	1
blockNumber	2
for(i=0;i<N;i++) {	
blockDataByte	1
}	
}	

ProtocolDiscriminator [HEX]	DsmccType [HEX]	MessageId [DEC]	DownloadId [DEC]	Reserved [HEX]	AdaptationLength [DEC]	MessageLength [DEC]	ModuleId [DEC]	ModuleVersion [DEC]	Reserved [HEX]	BlockNumber [DEC]
11	0x03	4099	2	FF	0	4072	1	1	FF	0
11	0x03	4099	2	FF	0	4072	1	1	FF	1
11	0x03	4099	2	FF	0	4072	1	1	FF	2
11	0x03	4099	2	FF	0	4072	1	1	FF	3
11	0x03	4099	2	FF	0	4072	1	1	FF	4
11	0x03	4099	2	FF	0	4072	1	1	FF	5
11	0x03	4099	2	FF	0	4072	1	1	FF	6
11	0x03	4099	2	FF	0	4072	1	1	FF	7
11	0x03	4099	2	FF	0	4072	1	1	FF	8
11	0x03	4099	2	FF	0	4072	1	1	FF	9
11	0x03	4099	2	FF	0	4072	1	1	FF	10
11	0x03	4099	2	FF	0	4072	1	1	FF	11
11	0x03	4099	2	FF	0	4072	1	1	FF	12
11	0x03	4099	2	FF	0	1282	1	1	FF	13

Secciones DSM-CC | Carrusel de Datos | Carrusel de Objetos 1 (file) | Carrusel de Objetos 2 (dir,srg ...

Figura 39. Reporte detallado de extracción del carrusel de datos

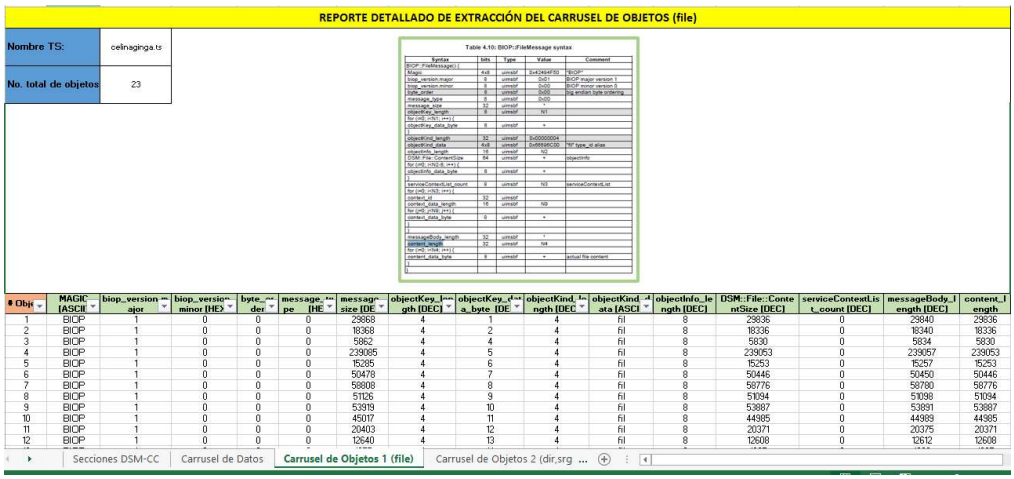


Figura 40. Reporte detallado de extracción del carrusel de objetos (fichero)

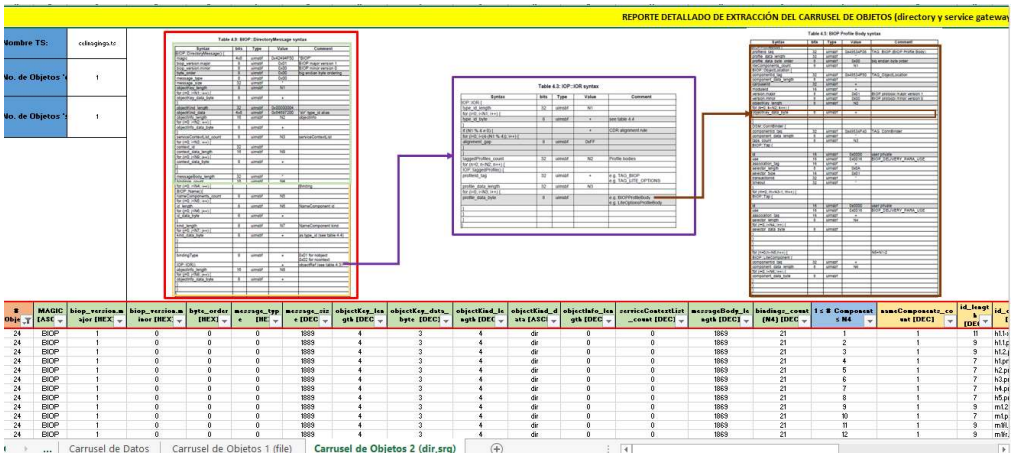


Figura 41. Reporte detallado de extracción del carrusel de objetos (directorio)

CAPÍTULO 4

4.1. Escenarios de Prueba y Resultados

Se comprobó el funcionamiento del software mediante la extracción exitosa de la información multimedia de 4 TS diferentes: *celinaginga.ts*, *espe_sek.ts*, *primeirojoao_ait.ts*, *EITV-GINGA.ts*, los cuales contaban con sección de datos DSM-CC. Los resultados de la extracción en relación del directorio de la aplicación reconstruida son mostrados a continuación, como base de los escenarios de prueba del proyecto.

4.1.1. Escenario 1: Construcción del directorio de la aplicación

El primer escenario de prueba del software extractor implementado es el de la construcción completa del directorio de la aplicación contenida en los TS mencionados anteriormente. Para esto es necesario poder visualizar el directorio y su contenido utilizando el explorador de Windows o cualquier explorador de archivos disponible que facilite dicha tarea, como por ejemplo el *Q-Dir* el cual es un software gratuito basado en el explorador de Windows común, pero con la ventaja de que el *Q-Dir* puede mostrar una vista multi-panel o multi-ventana que permite una rápida observación de un directorio y sus archivos, por lo que se decidió usar el *Q-Dir* en este escenario de prueba.

4.1.1.1 celinaginga.ts

El contenido (directorio y ficheros) de *celinaginga.ts* obtenidos después de la extracción se muestran en la figura 42, donde se puede apreciar que este TS contiene una aplicación de GINGA-NCL y su directorio se ve en (1), mientras que el contenido de la carpeta raíz "extracción_celinagingats" se ve en (2) y el de la carpeta "media" (3) que contiene 21 archivos multimedia de la aplicación entre imágenes (*.png) y archivos de texto (*.txt). Como se observa la carpeta raíz es la que en este caso contiene directamente los archivos "main.ncl" y "ConnectorBase.ncl" los cuales corresponden al código fuente de la aplicación.

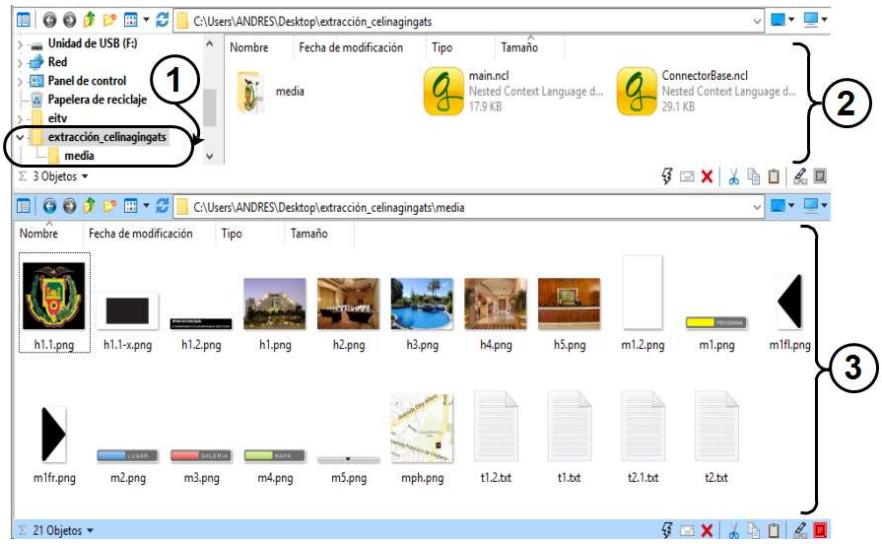


Figura 42. Directorio y ficheros recuperados de *celinaginga.ts*

4.1.1.2 espe_sek.ts

La figura 43 muestra los datos que se extrajeron del *espe_sek.ts* donde el directorio de la aplicación (1) en este caso es similar al de *celinaginga.ts* ya que dentro de su carpeta raíz "extracción_espe_sekts" (2) se encuentran los archivos del código fuente "main.ncl" y el "cBase.ncl" y además la carpeta "media" que contiene las imágenes (*.png) que constituyen el contenido multimedia de la aplicación mostrado en (3).

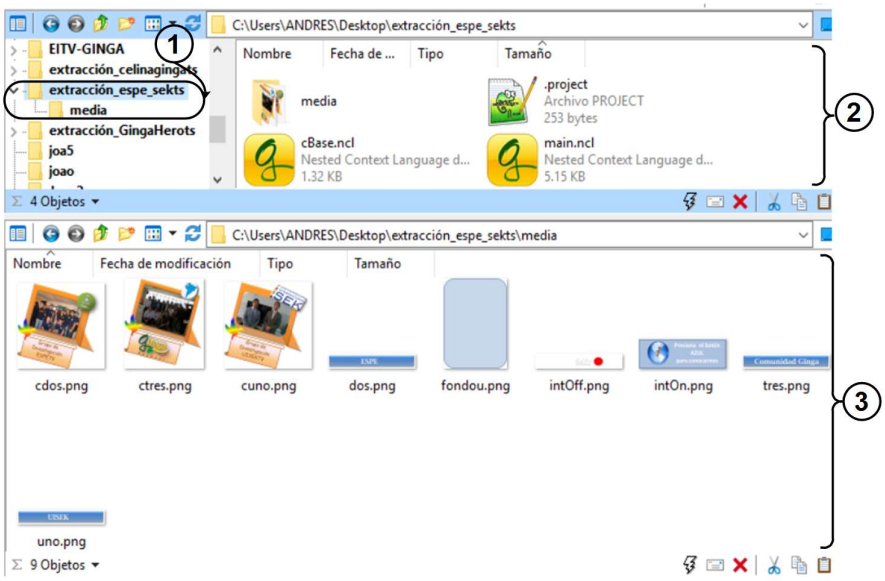


Figura 43. Directorio y ficheros recuperados de *espe_sek.ts*

4.1.1.3 primeirojoao_ait.ts

El *primeirojoao_ait.ts* contiene una aplicación desarrollada por Laboratorio TeleMidia PUC-Rio, que fue diseñada con el objetivo de la enseñanza de lenguaje NCL y cuyo tema fue inspirado en la vida del jugador de futbol Garrincha por lo que su contenido guarda relación con el futbol y el jugador como tal (Soares, 2010). La figura 44 muestra el contenido de la aplicación el cual se extrajo del TS mencionado anteriormente. Como se observa en (1) se tiene la estructura jerárquica del directorio y en (2) se puede observar que el directorio raíz "extracción_primeirojoaots" contiene únicamente la carpeta "Garrincha". En (3) se muestran los archivos del código fuente NCL ("GarrinchaFinal.ncl" y "newCausalConnBase.ncl") y la carpeta "media" que contiene 57 archivos mostrados en (4), los cuales conforman el contenido multimedia de la aplicación.

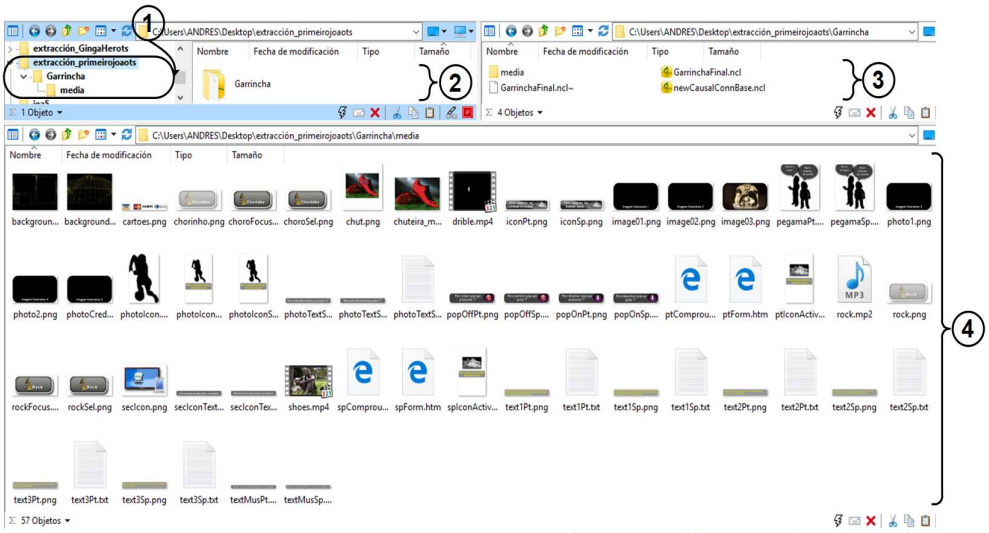


Figura 44. Directorio y ficheros recuperados de *primeirojoao_ait.ts*

4.1.1.4 EITV-GINGA.ts

El *EITV-GINGA.ts* es de propiedad privada de la cadena televisiva brasileña Rede GLOBO por lo que el contenido o programación de la aplicación interactiva que posee puede ser considerado confidencial. Por tal motivo en la figura 45 se muestra únicamente la estructura jerárquica del directorio (1) mas no su contenido extraído. La aplicación contenida dentro de dicho TS es NCL y LUA. Además se observa una

estructura jerárquica del directorio un poco más compleja (por número de carpetas contenidas) que el de las anteriores aplicaciones extraídas, por lo que se comprueba la importancia de poseer un algoritmo constructor que garantice una correcta organización del directorio y archivos independientemente de que tan simple o complejo sea su estructura.

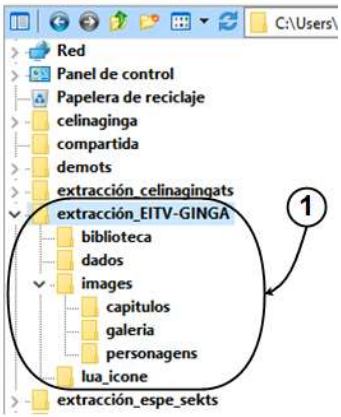


Figura 45. Directorio recuperado de *EITV-GINGA.ts*

4.1.2. Escenario 2: Ejecución de aplicación extraída

Aunque el objetivo principal del desarrollo del algoritmo y la de la implementación de *software* es la de extraer el directorio completo de la aplicación, se puede probar su correcta extracción mediante la ejecución de las mismas. A continuación en las figuras 46 y 47 se muestra la ejecución de las aplicaciones obtenidas de *celinaginga.ts* y *espe_sek.ts* respectivamente, usando el emulador de OPENGINGA (Quingaluisa, 2011).

Figura 46. Ejecución de aplicación de *celinaginga.ts*

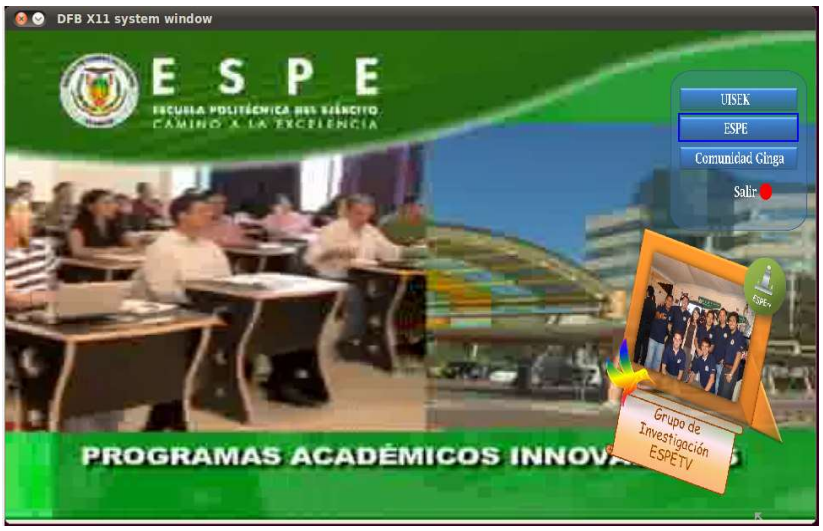


Figura 47. Ejecución de aplicación de *espe_sek.ts*

4.2. Análisis de datos estadísticos de extracción

Al culminar con el proceso de extracción de cada TS se obtuvo como resultado adicional su respectivo reporte resumido y detallado de extracción de donde se obtuvieron los siguientes resultados.

4.2.1. Reporte de extracción de celinaginga.ts

La tabla 5 muestra un reporte de las secciones DSM-CC extraídas de *celinaginga.ts*, es decir aquellas que contienen únicamente bloques de datos DDB, por

lo que en esta tabla se describe a cada módulo junto con el número total de secciones que los componen. Además también se visualiza datos como el del último número de la sección (campo *last section number*), el cual usualmente sumado 1, da como resultado el número total de secciones. También se observa la longitud de la última sección (*dsmcc section length*) la cual a diferencia de las demás secciones (que tienen una longitud común de 4093 bytes), siempre tendrá un tamaño menor.

Tabla 5.
Reporte Secciones DSM-CC de *celinaginga.ts*

Id. Módulo	No. Total Sec	No. Última Sec	Long. Última Sec [bytes]
1	14	13	1303
2	59	58	3296
3	4	3	3126
4	13	12	1725
5	15	14	1923
6	13	12	2373
7	14	13	1100
8	17	16	415
9	13	12	3479
10	6	5	3940
11	13	12	2543
12	1	0	2230

El reporte del carrusel de datos es muy similar al de las secciones DSM-CC dado que cada sección descargada representará a un bloque DDB. Por lo que para el carrusel de datos unicamente se presenta la figura 48, la cual muestra la comparación del tamaño de cada módulo en función del número de DDB's que cada uno posea, con respecto al tamaño total del carrusel de datos, destacando el módulo 2 por tener la mayor longitud y verificando que dado que la norma no limita el tamaño de los módulos, estos pueden variar mucho en cuanto a su capacidad.

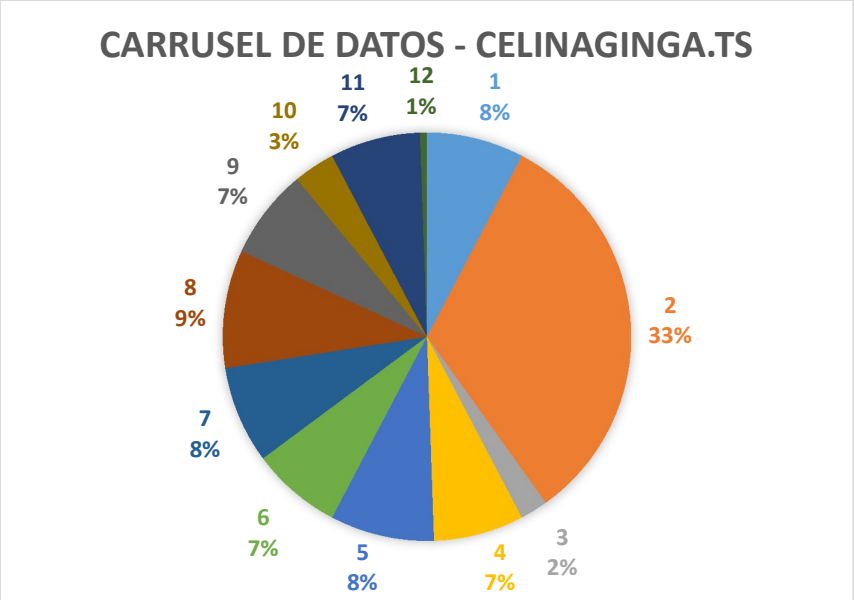


Figura 48. Tamaño de módulos de *celinaginga.ts*

El reporte del carrusel de objetos se muestra en las tabla 6 y 7, para objetos directorio y fichero respectivamente. La tabla 6 muestra los enlaces que describen los componentes de cada directorio, de acuerdo a su clave de objeto asociada. En la tabla 7 se muestra la relación de la clave de objeto con la de su correspondiente nombre de fichero. Esto a pesar de que los objetos tipo fichero en si no transportan el nombre del fichero, sin embargo por facilidad de visualización de los datos se decidió presentar la tabla de esa manera.

Tabla 6.
Reporte Carrusel de Objetos de *celinaginga.ts* (directorios)

Clave de Objeto	Nombre Carpeta	No. de Componentes	Claves de Objetos Asociados
3	media	21	4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24
0	..raíz..(srg)	3	1,2,3

Tabla 7.
Reporte Carrusel de Objetos de *celinaginga.ts* (ficheros)

Clave Objeto	Nombre Fichero	Tamaño Fichero [bytes]
4	h1.1-x.png	5830
5	h1.1.png	239053
6	h1.2.png	15253
7	h1.png	50446
8	h2.png	58776
9	h3.png	51094
10	h4.png	53887
11	h5.png	44985
12	m1.2.png	20371
13	m1.png	12608
14	m1fl.png	1825
15	m1fr.png	1739
16	m2.png	18168
17	m3.png	17684
18	m4.png	16766
19	m5.png	7389
20	mph.png	50393
21	t1.2.txt	32
22	t1.txt	50
23	t2.1.txt	376
24	t2.txt	237
1	ConnectorBase.ncl	29836
2	main.ncl	18336

El análisis de las tablas y figuras del reporte de extracción del *espe_sek.ts*, *primeirojoao_ait.ts* e *EITV-GINGA.ts* se lo realiza de la misma manera que para el *celinaginga.ts* por lo que a continuación se presentan únicamente los datos de cada extracción.

4.2.2. Reporte de extracción de espe_sek.ts

Tabla 8.
Reporte Secciones DSM-CC de *espe_sek.ts*

Id. Módulo	No. Total Sec	No. Última Sec	Long. Última Sec [bytes]
1	2	1	2977
2	162	161	212
3	171	170	3718
4	151	150	3885
5	7	6	1218
6	18	17	2746
7	9	8	135
8	1	0	1259

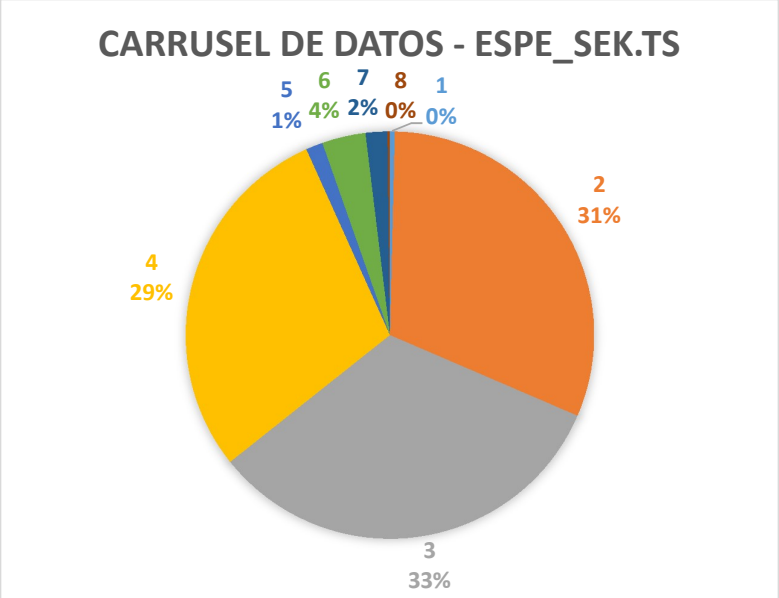


Figura 49. Tamaño de módulos de *espe_sek.ts*

Tabla 9.
Reporte Carrusel de Objetos de *espe_sek.ts* (directorios)

Clave de Objeto	Nombre carpeta	No. de Componentes	Claves de Objetos Asociados
4	media	9	5,6,7,8,9,10,11,12,13
0	..raíz..(srg)	4	1,2,3,4

Tabla 10.
Reporte Carrusel de Objetos de *espe_sek.ts* (ficheros)

Clave Objeto	Nombre Fichero	Tamaño Fichero [bytes]
5	cdos.png	654767
6	ctres.png	694867
7	cuno.png	613714
8	dos.png	11250
9	fondou.png	8484
10	intOff.png	5721
11	intOn.png	71797
12	tres.png	19834
13	uno.png	12714
1	.project	253
2	cBase.ncl	1355
3	main.ncl	5276

4.2.3. Reporte de extracción de primierojoao_ait.ts

Tabla 11.
Reporte Secciones DSM-CC de *primeirojoao_ait.ts*

Id. Módulo	No. Total Sec	No. Última Sec	Long. Última Sec [bytes]
1	10	9	1683
2	38	37	1368
3	106	105	2647
4	5	4	3199
5	18	17	3682
6	19	18	76
7	19	18	76
8	78	77	579
9	35	34	2757
10	14	13	1510
11	35	34	2242
12	34	33	2435
13	48	47	1858
14	45	44	2311
15	30	29	1715
16	50	49	2612
17	50	49	1741

Continua ➔

18	38	37	1022
19	36	35	819
20	35	34	2065
21	26	25	855
22	9	8	1481
23	11	10	787
24	21	20	1215
25	22	21	398
26	20	19	1512
27	19	18	3654
28	1	0	3105
29	17	16	1927
30	218	217	1888
31	18	17	3509
32	18	17	3002
33	18	17	3002
34	43	42	3499
35	2	1	3244
36	136	135	1841
37	1	0	3123
38	17	16	2378
39	16	15	3372
40	7	6	3136
41	2	1	1901

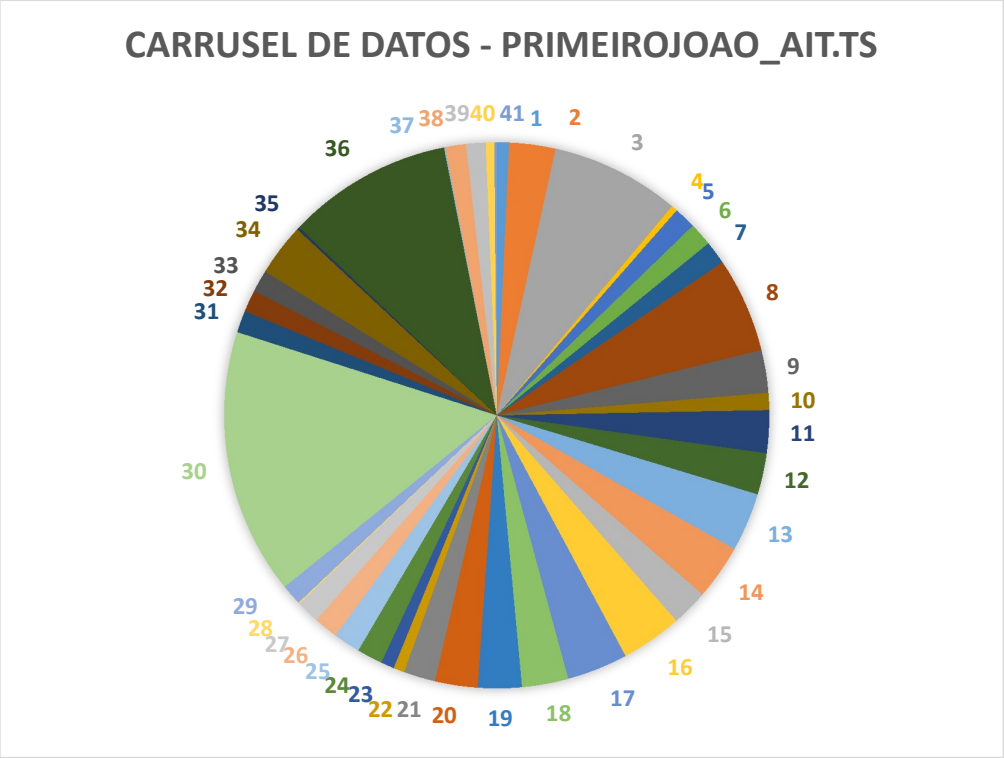


Figura 50. Tamaño de módulos de *primeirojoao_ait.ts*

Tabla 12.
Reporte Carrusel de Objetos de *primeirojoao_ait.ts* (directorios)

Clave de Objeto	Nombre carpeta	No. de Componentes	Claves de Objetos Asociados
4	media	57	5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61
1	Garrincha	4	2,3,4,62
0	..raiz..(srg)	1	1

Tabla 13.
Reporte Carrusel de Objetos de *primeirojoao_ait.ts* (ficheros)

Clave Objeto	Nombre Fichero	Tamaño Fichero [bytes]
5	background.png	151739
6	background2.png	429506
7	cartoes.png	19392
8	chorinho.png	72733

Continua ➔

9	choroFocus.png	73193
10	choroSel.png	73193
11	chut.png	313590
12	chuteira_mod.png	140930
13	drible.mp4	54297
14	iconPt.png	140415
15	iconSp.png	136542
16	image01.png	192889
17	image02.png	181144
18	image03.png	119558
19	pegamaPt.png	201775
20	pegamaSp.png	200904
21	photo1.png	151393
22	photo2.png	143058
23	photoCred.png	140238
24	photoIcon.png	102434
25	photoIconPt.png	33938
26	photoIconSp.png	34069
27	photoTextSinglePt.png	3613
28	photoTextSingleSp.png	3529
29	photoTextSp.txt	33
30	popOffPt.png	82464
31	popOffSp.png	85713
32	popOnPt.png	78695
33	popOnSp.png	76771
34	ptComprou.htm	1120
35	ptForm.htm	1870
36	ptIconActive.png	66912
37	rock.mp2	884139
38	rock.png	72560
39	rockFocus.png	72053
40	rockSel.png	72053
41	secIcon.png	174200
42	secIconTextPt.png	3701
43	secIconTextSp.png	3494
44	shoes.mp4	550680
45	spComprou.htm	1138

Continua ➔

46	spForm.htm	1870
47	spIconActive.png	67363
48	text1Pt.png	15609
49	text1Pt.txt	172
50	text1Sp.png	16220
51	text1Sp.txt	183
52	text2Pt.png	9786
53	text2Pt.txt	93
54	text2Sp.png	11187
55	text2Sp.txt	125
56	text3Pt.png	10426
57	text3Pt.txt	94
58	text3Sp.png	9593
59	text3Sp.txt	92
60	textMusPt.png	5205
61	textMusSp.png	5130
2	GarrinchaFinal.ncl	19104
3	GarrinchaFinal.ncl~	19058
62	newCausalConnBase.ncl	7265

4.2.4. Reporte de extracción de EITV-GINGA.ts

Tabla 14.
Reporte Secciones DSM-CC de *EITV-GINGA.ts*

Id. Módulo	No. Total Sec	No. Última Sec	Long. Última Sec [bytes]
12	16	15	2506
13	10	9	3547
14	9	8	3213
15	10	9	3431
16	17	16	1589
17	14	13	1698
18	10	9	376
19	15	14	1924
20	11	10	2826
21	11	10	792
22	8	7	3890
23	11	10	1164

Continua ➔

24	9	8	2814
25	11	10	1226
26	16	15	2717
27	15	14	736
28	23	22	35
29	15	14	60
30	4	3	675
1	16	15	817
2	13	12	269
3	4	3	819
4	16	15	666
5	16	15	2706
6	16	15	3767
7	16	15	2000
8	16	15	1236
9	15	14	3393
10	15	14	1392
11	15	14	3109

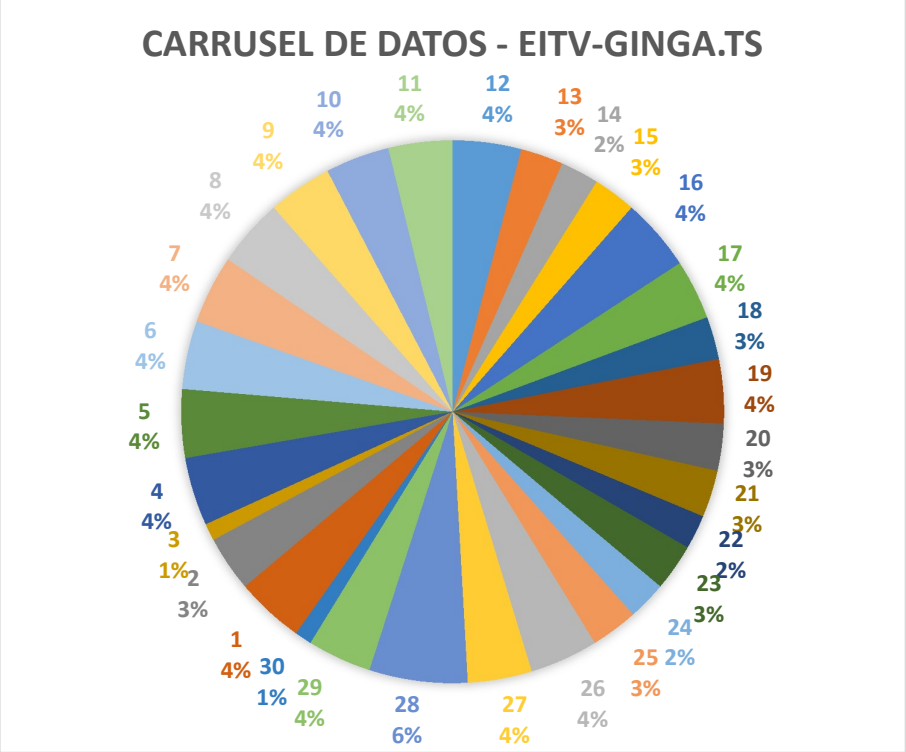


Figura 51. Tamaño de módulos de *EITV-GINGA.ts*

Tabla 15.
Reporte Carrusel de Objetos de *EITV-GINGA.ts* (directorios)

Clave de Objeto	Nombre carpeta	No. de Componentes	Claves de Objetos Asociados
2	biblioteca	6	3,4,5,6,7,8
10	dados	4	11,12,13,14
44	capitulos	17	45,46,47,48,49,50,51,52,53,54,55,56,57,58,59,60,61
62	galeria	12	63,64,65,66,67,68,69,70,71,72,73,74
92	personagens	20	93,94,95,96,97,98,99,100,101,102,103,104,105,106,107,108,109,110,111,112
18	images	55	19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,62,75,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,92,113,114,115,116,117,118,119,120,121,122
123	lua_icone	3	124,125,126
0	..raiz..(srg)	18	1,2,9,10,15,16,17,18,123,127,128,129,130,131,132,133,134,135

4.2.5. Reporte general de extracción

La tabla 16 muestra un reporte general de la extracción de Secciones DSM-CC, carrusel de datos y carrusel de objetos de los 4 TS's de prueba.

Tabla 16.
Datos generales de extracción

Archivo TS	*celinaginga.ts*	*espe_sek.ts*	*primeirojoao_ait.ts*	*EITV-GINGA.ts*
PID del DSM-CC	2004	2004	900	1001
No. de Módulos	12	8	41	30
No. Sec DSM-CC	183	522	1416	462
No. de DDB	182	521	1385	393
No. de DII	1	1	31	69
No. Objetos "srg"	1	1	1	1
No. Objetos "dir"	1	1	2	7
No. Objetos "fil"	23	12	60	128
Tiempo de extracción (seg)	36	169	243	679

Entre los datos presentados en la tabla 16, se encuentra el PID del DSM-CC el cual fue encontrado en la PMT de cada TS. Por otro lado se muestra también el número de módulos descargados (correspondientes a un ciclo del carrusel de datos) a partir de la lectura y desencapsulación de las secciones DSM-CC. Como se observa la cantidad de módulos extraídos no necesariamente refleja una mayor o menor cantidad de secciones encontradas, como por el ejemplo el *espe_sek.ts* que posee únicamente 8 módulos tendrá más secciones que EITV-GINGA.ts, a pesar de tener 30 módulos. Esto sin contar que el número de secciones que muestra la tabla cuenta tanto las secciones con *table id* de 0x3C (datos) como con 0x3B (control) por lo que la diferencia entre el número de secciones de datos de *espe_sek.ts* y *EITV-GINGA.ts* puede ser mayor, cosa que se demuestra al revisar en el número de DDB's descargados y el número de DII localizados de los dos TS's mencionados.

En lo que respecta a número bloques de control de descarga DII, *celinaginga.ts* y *espe_sek.ts* cumplen con lo mínimo necesario que dicta la norma ISO 13818-6, ya que solo contienen un bloque de control en un ciclo del carrusel de datos mientras que al observar el número de DII de *primeirojoao_ait.ts* y *EITV-GINGA.ts* se comprueba que el número o frecuencia con la que estos bloques DII son introducidos en el carrusel no está restringida por la norma, pero con la condicionante de que mientras más veces se introduzcan bloques de control en un solo ciclo del carrusel también el tiempo de extracción aumentará debido a que esto resta ancho de banda a los DDB's que son los que llevan la carga útil (datos de la aplicación). Esto se demuestra ya que aunque el *primeirojoao_ait.ts* contiene más módulos (41) y DDB's (1385) que el *EITV-GINGA.ts* (30 módulos y 393 DDB's), al poseer este último un número mayor de bloques DII (según la tabla 16), la descarga se vuelve más lenta (679 seg), siendo su tiempo de extracción casi 3 veces mayor de lo que se demora la extracción del *primeirojoao_ait.ts* (243 seg).

Los datos relacionados con el carrusel del de objetos que se muestran en la tabla 16 hacen constatar el concepto del uso del objeto puerta de enlace ("srg"), el cual establece que únicamente habrá un solo objeto de este tipo dentro del carrusel de

objetos ya que este representa al directorio raíz. Por otra parte el número de objetos directorio (“dir”) se relaciona de forma directa con el número de carpetas que contenga el directorio (excepto la raíz) de la aplicación.

CAPÍTULO 5

5.1. Conclusiones

- Para elaborar el software de extracción de datos de un TS con la especificación de MPEG-2 se utilizó los conceptos de secciones DSM-CC, carrusel de datos, carrusel de objetos, los cuales constituyen protocolos que conforman el perfil de descarga, control y organización de la información.
- Entre algunos métodos para visualizar la información que transportan los mensajes U-N DSM-CC de un TS, se encuentra el uso de un *set-top box* que posea el *middleware* adecuado (GINGA-NCL) para ejecutar la aplicación interactiva o también utilizando un *software* que sirva de analizador de TS (DemuxToyLite, TSReader, etc). Sin embargo estos métodos constituyen una forma limitada e insuficiente de visualización de los datos ya que no permiten un análisis detallado de los mismos, al realizar únicamente la ejecución de la aplicación o simplemente indicar las tablas PSI/SI que describen la configuración del DSM-CC en el TS.
- El carrusel de datos por la naturaleza cíclica de su transmisión permite al cliente descargar todos los datos en cualquier momento en el que el TS se esté receptando, sin embargo la estructura de este no provee un forma de ordenamiento de la información de acuerdo a un sistema jerárquico de archivos y directorio por lo que se debe hacer uso del concepto de carrusel de objetos. También el carrusel de datos necesita ser encapsulado en secciones DSM-CC las cuales le permiten ensamblar los bloques de datos y de control dentro de los paquetes del TS, además de proveer una forma de detección de errores mediante CRC-32.
- La identificación y discriminación de paquetes que transportan los mensajes U-N DSM-CC, se basa en el análisis del descriptor de flujo elemental localizado en la PMT cuyo identificador de tipo de flujo será 0x0B.
- La extracción de las secciones DSM-CC se puede llevar a cabo siguiendo las etapas de identificación del PID de los paquetes del DSM-CC, detección del inicio de descarga, localización de al menos un bloque de control DII para encontrar el número de módulos que se deben descargar y extraer todas las secciones que compongan cada uno de los módulos considerando los casos especiales del uso del

pointer field para eliminar los bytes de relleno al desencapsular cada bloque DDB y teniendo como resultado final un fichero temporal que contenga un único ciclo del carrusel de datos.

- El cálculo de paquetes intermedios entre el inicio y final de una sección (*paquetes_sec*) permite optimizar el tiempo de extracción de las secciones al evitar chequear el estado del bit de indicador de arranque de la cabecera de todos los paquetes del TS, por lo que esto se puede considerar como un aporte en cuanto a mejorar la eficiencia del proceso de descarga de datos se refiere.
- El carrusel de datos de una capa se caracteriza por no tomar en cuenta el uso del bloque de control DSI en donde este, aun cuando es incluido en la transmisión de carrusel solo será de manera conceptual, con el objetivo conformar una estructura general del carrusel de datos que describe a un solo grupo de módulos.
- El algoritmo de desencapsulación del carrusel de datos se basa principalmente en la eliminación de la cabecera de cada DDB y almacenamiento de su carga útil en un nuevo fichero temporal en donde se termine formando el carrusel de objetos.
- El carrusel de objetos se basa en la transmisión de objetos tipo fichero (“fil”), directorio (“dir”) y puerta de servicio (“srg”), los cuales contienen la información de cada fichero y una serie de enlaces que permite que en el lado del receptor se pueda relacionar a cada fichero con su respectivo nombre y a su vez estos puedan ser ubicado el carpeta que corresponda, en relación al directorio raíz de la aplicación.
- La desencapsulación del carrusel de objetos y organización de la información se puede llevar a cabo realizando un análisis e identificación de todos los objetos en el carrusel, para clasificarlos y almacenarlos según corresponda mediante el sistema de vectores propuesto (“objeto”, “nombre carpeta” y “nombre fichero”), con el propósito de relacionar las claves de objeto para construcción completa del directorio y ubicación de cada fichero dentro del mismo
- El algoritmo de extracción y construcción de datos desarrollado en este trabajo, resulta muy eficiente y sencillo, dado que no necesita la utilización de muchos de los parámetros que tanto las secciones DSM-CC, carrusel de datos y objetos presentan en sus respectivas estructuras, para cumplir con el objetivo que es de

recuperar la información de una aplicación de TDT en forma ordenada en su respectivo directorio.

- La aplicación que se le puede dar a al algoritmo propuesto o al software que se pueda desarrollar con este es variada, en especial en países de América Latina donde la TDT, aún sigue en proceso de implementación, lo que genera que se deba hacer pruebas de transmisión de los diferentes servicios que la TDT posee. Esto significa que en lo que respecta a la transmisión de aplicaciones interactivas y datos en general, se deba tener formas más sencillas de recuperar la información.
- Conceptos desarrollados y propuestos en este proyecto tales como el cálculo de *paquetes_sec*, uso de "carpetas finales" para generar rutas del directorio y el sistema de vectores de almacenamiento temporal, pueden ser considerados como un aporte para mejorar la eficiencia en cuanto al proceso de descarga, control y organización de aquellos datos que son transportados bajo la especificación de MPEG-2 TS.

5.2. Recomendaciones

- Durante la extracción de las secciones DSM-CC es importante considerar e interpretar bien los casos especiales del uso del *pointer field*, ya que esto permitirá descartar todos los bytes de relleno (0xFF) presentes. Recordando que un solo byte de relleno que quede sin eliminar durante este proceso, será suficiente para que las etapas de extracción del carrusel de datos y de objetos no se realicen de forma satisfactoria.
- Dado que el carrusel de objetos no transmite cada uno de sus objetos en un orden particular, es importante señalar que la implementación de un sistema de vectores semejante al propuesto en este proyecto debe ser utilizado para relacionar la información recopilada de cada objeto una vez finalizada en proceso de análisis de todo el carrusel.

5.3. Trabajos Futuros

- En el presente trabajo no es considerado ninguno de los códigos detectores de errores (CRC-32 o *Checksum*) que posee la estructura de las secciones DSM-CC,

por lo que sería interesante realizar un trabajo en cuanto a la comparación de datos extraídos cuando se descarta o no bloques de datos con información errónea.

- La implementación del algoritmo desarrollado con diferentes lenguajes de programación como Java, Python, etc., resultaría interesante para poder comparar el rendimiento que cada uno posee en cuanto al tiempo de extracción de un mismo TS, además de poder tener a disposición un software multiplataforma para realizar dicha tarea.

5.4. Publicaciones

Núñez, A, Olmedo, G. Villamarín, D. (2016). *Design, implementation and evaluation of data carrousel extractor algorithm on MPEG-2 TS for Digital Terrestrial Television*. CLEI 2016: Latin American Symposium on Infrastructure, Hardware and Software. IEEE.

REFERENCIAS

ABNT NBR. (2008). Televisión digital terrestre. *Codificación de datos y especificaciones de transmisión para radiodifusión digital Parte 3: Especificación de transmisión de datos*. 15603-3.

ABNT NBR. (2009). Televisión digital terrestre. *Multiplexación y servicios de información (SI) Parte 2: Estructura de datos y definiciones de la información básica de SI*. 15603-2.

Altais Digital Ltd. (15 de Mayo de 2015). *DemuxToy Lite - DVB Transport Stream Analyser.* Obtenido de Digital Digest: http://www.digital-digest.com/software/demuxtoy.html

ARCOTEL. (1995). *Ley de Radiodifusión y Televisión. Registro Oficial No. 691. 1995.* Obtenido de http://www.telecomunicaciones.gob.ec/wp-content/uploads/downloads/2012/11/Ley-de-Radiodifusion-y-Television.pdf

ARCOTEL. (1996). *Reglamentos a la Ley de Radiodifusión y Televisión. Registro Oficial Suplemento 864.* Obtenido de http://www.telecomunicaciones.gob.ec/wp-content/uploads/downloads/2012/11/Reglamento-de-Ley-de-Radiodifusion-y-Television.pdf

Benavides, N. (2015). Desfragmentador del flujo de transporte (TS) y Desfragmentador del flujo de transporte (TS) y. Obtenido de http://repositorio.espe.edu.ec/handle/21000/10434.

Cho, J. S. (2008). A new content-related advertising model for interactive television. In Broadband Multimedia Systems and Broadcasting. *2008 IEEE International Symposium* (págs. 1-9). IEEE.

Cool.Stf. (2014). *MPEG-2 Transport Stream Analysis and Recording*. Obtenido de TS Reader: http://www.coolstf.com/tsreader/

Crinon, R. J. (1997). The DSM-CC Object Carousel for broadcast data. *Digest of Technical Papers. ICCE (International Conference)* (págs. 246-247). IEEE.

Dos Santos Jr, J. B. (2012). Towards interactivity for citizenship: An approach to interactive digital television. *In Consumer Communications and Networking Conference (CCNC), 2012 IEEE* (págs. 723-728). IEEE.

Ether Guide Systems. (2016). *pointer field [MPEG-2 Semantics]*. Recuperado el 20 de Abril de 2016, de http://www.etherguidesystems.com/help/sdos/mpeg/semantics/mpeg-2/pointer field.aspx.

ETSI EN. (2004). Digital Video Broadcasting (DVB); DVB specification for data broadcasting. European Broadcasting Union.

ETSI TR. (2003). Digital Video Broadcasting (DVB). *Implementation guidelines for Data Broadcasting*.

Guobin, W. H. (2006). Advanced Software Architecture for Processing Bulk Carousel Data in a Data Broadcasting Push Service. *In 2006 8th international Conference on Signal.*

Hamada, I. M. (2005). *Washington, DC: U.S. Patente nº 6,931,198.*

Haubold, A. (2007). Matlab for first-year college engineers. *37th Annual Frontiers In Education Conference - Global Engineering: Knowledge Without Borders, Opportunities Without Passports*, (págs. F1H-7-F1H-12). Milwaukee, WI.

ISO/IEC. (Septiembre de 1998). Generic coding of moving pictures and associated audio information - Part 6: Extensions for DSM-CC. 13818-6.

ISO/IEC. (2015). Information technology. *Generic coding of moving pictures and associated audio information – Part 1: Systems*.

Ministerio de Telecomunicaciones y Sociedad de la Información. (2010). *Televisión Digital Terrestre en el Ecuador*. Obtenido de http://www.telecomunicaciones.gob.ec/television-digital-terrestre-en-el-ecuador/

Morris, S. (2011). *How To Become An Expert In DSM-CC*. Obtenido de TV Without Borders: http://www.interactivetvweb.org/tutorials/dtv_intro/dsmcc

Park, D. H. (2006). Real-time carousel caching and monitoring in data broadcasting. *In Consumer Electronics, 2006. ICCE'06. 2006 Digest of Technical Papers* (págs. 273-274). IEEE.

Quingaluisa, A. T. (2011). Estudio e investigación del MIDDLEWARE GINGA J del estándar brasileño de televisión digital. Caso práctico: desarrollo de una aplicación interactiva aplicando metodología OPENUP/BASIC como parte del

Proyecto ESPE-GINGA. Obtenido de http://repositorio.espe.edu.ec/bitstream/21000/4748/1/T-ESPE-032865.pdf.

Soares, L. C. (2010). *Garrincha e o Primeiro João*. Obtenido de Laboratório TeleMídia da PUC-Rio: http://clube.ncl.org.br/node/46

Villamarin, D. I. (2013). Generating a transport stream for digital terrestrial television system in conformance with ISDB-Tb standard. *In Communications and Computing (COLCOM), 2013 IEEE Colombian Conference* (págs. 1-5). IEEE.

Xiang, L. &. (2009). Influence of Interactive Television on Electronic Commerce. In Management of e-Commerce and e-Government. *ICMECG'09. International Conference* (págs. 513-516). IEEE.

Zhang, H. J. (2004). Design and implementation of broadcast file system based on DSM-CC data carousel protocol. *Consumer Electronics, IEEE Transactions*, 929-933.

Zuffo, M. K. (2008). TV digital aberta no Brasil. *Políticas estruturais para um modelo nacional*. São Paulo. Obtenido de http://blog.joaomattar.com/2007/07/08/gestao-estrategica-de-negocios-editoriais.

ANEXOS