UNIVERSIDAD  DE VALLADOLID

E.T.S.I. TELECOMUNICACIÓN

TRABAJO FIN DE MÁSTER

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

Desarrollo de una solución de vídeo vigilancia IP

Autor:
D. Javier García Cartón

Tutor:
D. Míriam Antón Rodríguez

Valladolid, julio de 2016

TÍTULO:	**Desarrollo de una solución de vídeo vigilancia IP**
AUTOR:	**D. Javier García Cartón**
TUTOR:	**Dña. Míriam Antón Rodríguez**
DEPARTAMENTO:	**Teoría de la Señal y Comunicaciones e Ingeniería Telemática**

TRIBUNAL

PRESIDENTE:	**Dña. Mª Ángeles Pérez Juárez**
VOCAL:	**D. Javier Aguiar Pérez**
SECRETARIO	**D. David González Ortega**
SUPLENTE	**D. Manuel Rodríguez Cayetano**
SUPLENTE	**D. Jaime Gómez Gil**
SUPLENTE	**Dña. María García Gadañón**

FECHA:	**Julio de 2016**
CALIFICACIÓN:	

RESUMEN DEL TFM

Hoy en día está muy extendido el uso de transporte urbano público, tanto autobús como tren o tranvía. Estos transportes, además de funcionar correctamente, deben garantizar la seguridad de los pasajeros. En la situación actual que se está viviendo a nivel mundial, cobra especial importancia esta seguridad en los grandes medios de transporte.

Existen diversos mecanismos para garantizar la seguridad de los viajeros, uno de los cuales es el empleo de cámaras. Estos sistemas tienen principalmente una función disuasoria para evitar posibles delitos, garantizando el bienestar de los pasajeros. Otra función de este tipo de sistemas es, por ejemplo, el control de la cantidad de viajeros, asegurar que las zonas de entrada y salida de pasajeros están despejadas, prevenir comportamientos inadecuados o servir de evidencia en caso de incidentes o accidentes.

Sin embargo, tradicionalmente este tipo de vídeo vigilancia se ha realizado con sistemas que sólo permitían el visionado directo en un circuito cerrado y grababan los vídeos en dispositivos de almacenamiento conectados físicamente al circuito cerrado. De esta manera, en caso en caso de producirse un accidente grave, se destruirían los sistemas de almacenamiento, ya sea un sistema ubicado en una tienda, un banco, un espacio público o un transporte urbano público.

En este trabajo fin de máster se pretende solventar los problemas de este tipo de vídeo vigilancia. Para ello se desarrollará un software para la captura de vídeo de las cámaras a través de una red IP. Además, este tipo de redes permite reaccionar a eventos detectados por la cámara, permitiendo una integración con diversos servicios. Asimismo, este software permite prescindir de programas independientes para cada marca de cámaras, unificando una gran variedad en una sola aplicación.

PALABRAS CLAVE

Vídeo vigilancia, cámara IP, Everfocus, RTP, H.264.

ABSTRACT

Nowadays, the use of urban public transport is widespread, either bus or train and tram. These transports not only must work correctly, but also guarantee passengers' safety. Particularly in today's context, security is one of the major concerns for the principal means of transport.

One of the diverse means to guarantee passengers' security is the use of surveillance cameras. These systems have a dissuasive purpose to avoid crimes and, in that way, they guarantee passengers' welfare. Other roles of these systems are, for instance, to control the flow of travellers, ensure that entrance/exit areas are clear, prevent inappropriate behaviours or provide video footage in case of incidents or accidents.

However historically, the systems used for this kind of video surveillance only allow live viewing in closed-circuit and video recording in storage devices, which are physically connected to that closed-circuit. In this way, in case of a serious accident, storage devices will be destroyed, no matter if it is a system based in a shop, a bank, a public space or in an urban public transport.

This Master's Degree Final Project proposes solutions to the issues of this kind of video surveillance through the development of a software, which captures the video footage from the cameras by means of an IP network. In addition, this kind of networks allows to act accordingly to certain events detected by the camera, being possible the integration into diverse services. Likewise, this software enables to dispense with specific programs for each camera brand, unifying a wide diversity into just one application.

KEYWORDS

Video surveillance, IP camera, Everfocus, RTP, H.264

Agradecimientos

En estas líneas me gustaría agradecer a todas aquellas personas que me han acompañado y ayudado a llegar hasta aquí.

En primer lugar a mis padres y hermanos, quienes me han apoyado en todo momento a lo largo de estos años de carrera.

En segundo lugar agradecer a todos mis amigos, esa gente que he conocido a lo largo de estos años y con quien he compartido buenos y no tan buenos momentos, no voy a citar nombre porque la lista es bastante extensa. Desde aquí, sólo expresar mi deseo de que podamos seguir en contacto recordando las historias vividas juntos y viviendo otras nuevas.

Doy las gracias también a los tutores de este proyecto, Miriam y Javier, por ayudarme en el desarrollo de este proyecto resolviendo mis dudas. Agradecerles también el esfuerzo realizado en la revisión línea a línea de esta memoria.

Índice general

Capítulo 1

Introducción

1.1. Motivación

En la actualidad cada vez adquiere mayor importancia la seguridad de las personas en nuestra sociedad. Resulta especialmente importante poder garantizar la seguridad en los lugares públicos y generar sensación de seguridad en las personas, a lo cual contribuye el auge que sigue teniendo lugar en el sector de las telecomunicaciones.

Dicha necesidad de seguridad se puede observar en los grandes espacios públicos. En la gran mayoría de espacios públicos donde se producen grandes concentraciones de personas, es habitual encontrarse con cámaras de vídeo vigilancia que permitan detectar incidentes y resolverlos en el menor tiempo posible. Hoy en día una solución es la utilización de sistemas de vídeo vigilancia analógicos en una red cerrada, realizando la visualización y grabación en equipos de la propia red.

La eficacia de este sistema se halla en su capacidad de poder controlar múltiples zonas al mismo tiempo y permitir a los cuerpos de seguridad actuar rápida y coordinadamente, además de crear sensación de seguridad entre las personas.

Sin embargo, este tipo de comunicación conlleva una serie de desventajas que este trabajo trata de solucionar. Entre ellas cabe destacar:

- En primer lugar, la transmisión es analógica, por lo que se deben utilizar equipos de grabación analógicos o volver a digitalizar la señal para almacenar el vídeo.

- En segundo lugar, las señales analógicas son menos seguras y cualquiera con acceso a la infraestructura cableada puede interceptar la señal y/o ver el vídeo de las cámaras analógicas.

- En tercer lugar, es necesario digitalizar la señal y utilizar un servidor de vídeo para permitir el acceso remoto y otras funciones que presentan las cámaras IP.

Con este Trabajo Fin de Máster, realizado durante la beca en RSS (Rail & Station Systems Section), GMV [1], se pretende desarrollar una solución para la grabación de vídeo a partir de cámaras IP y su configuración previa. Además se busca tener una solución que sea compatible con múltiples modelos de cámaras IP.

1.2. Objetivos

El objetivo fundamental de este trabajo es el **desarrollo de una solución de vídeo vigilancia IP, partiendo de soluciones previas, para nuevos modelos de cámaras que permita la grabación de vídeo con características configurables**. Los requisitos para alcanzar este objetivo son:

- La necesidad de que la solución sea compatible con las cámaras Everfocus [2].

- Los distintos ficheros de vídeo en los que se divide la grabación deben estar temporalmente sincronizados entre sí, sin que haya pérdidas entre uno y otro.

- En el nombre de cada fichero de vídeo debe aparecer la fecha y la hora con segundos en la que comienza el vídeo.

- La configuración de vídeo de las cámaras debe realizarse al lanzar la aplicación, antes de comenzar la grabación de vídeo.

- La solución ha de funcionar de manera equivalente al implantado hasta el momento, pero incluyendo la compatibilidad con el nuevo modelo de cámara.

Con este nuevo software se persigue solventar los principales problemas presentes en las soluciones que emplean cámaras analógicas y que se han detallado en el apartado anterior.

Además de solucionarse los problemas de los sistemas que utilizan cámaras analógicas, se logra la compatibilidad con un modelo de cámara de otro fabricante adicional.

1.3. Contextualización

GMV vende soluciones de diversos tipos a clientes como pueden ser Renfe [3], Talgo [4], Renault [5] o Auvasa [6], entre muchos otros. Una de estas soluciones es la vídeo vigilancia a través de cámaras IP.

Este sistema de vídeo vigilancia se emplea en trenes, tranvías y autobuses para controlar el estado de las puertas, la cantidad de viajeros, el exterior del vehículo y detectar o grabar incidentes.

El sistema funciona sobre una red digital, por lo que es posible integrarlo con otros sistemas como megafonía, vídeo o localización.

La flexibilidad de este sistema posibilita que se utilice en entornos diferentes al transporte público, como podrían ser estaciones, tiendas, bancos, hogares, espacios públicos, etc. En estos entornos se podrían utilizar funcionalidades como la detección de movimiento para generar alertas y llevar a cabo acciones o el acceso remoto desde el dispositivo del cliente.

1.4. Fases y métodos

En una primera fase de la realización del presente trabajo es necesario un periodo de documentación sobre la solución existente hasta el momento y su funcionamiento con el modelo de cámaras Moxa [7].

En una segunda fase, debido a la necesidad de configurar las cámaras antes de comenzar la grabación de vídeo, se realiza su configuración manual y se estudian los manuales de las cámaras para generar las peticiones de configuración en el código.

La siguiente fase aborda el desarrollo del código necesario para la grabación del vídeo generado por los modelos de cámaras Everfocus.

Por último, se realizan pruebas de grabación de vídeo, con el objetivo de corregir los posibles fallos y errores de la solución.

1.5. Medios

Las herramientas utilizadas para el desarrollo de las tareas propuestas en el apartado anterior son:

Software:

- Eclipse [8]: Entorno de desarrollo de aplicaciones C/C++.
- Ubuntu 10.04 [9]: Es una distribución Linux basada en *Debian GNU/Linux* [10].
- MP4 Reader [11]: Herramienta de análisis del contenido de ficheros con formato de vídeo.
- Wireshark [12]: Analizador de protocolos de red.
- HxD [13]: Editor hexadecimal.

Hardware:

- Ordenador:
  - Procesador: Intel® Core ™ i5-4590 CPU a 3.30GHz.
  - 8GB de memoria RAM.
  - 500GB de disco duro.
  - Tarjeta gráfica: Intel® HD Graphics 4600.

- Cámara Everfocus EMN2120 nevio HD Series [14]: Cámara IP utilizada para el desarrollo de la solución de vídeo vigilancia.

1.6. Estructura de la memoria

A continuación se define la distribución de los capítulos del resto de este documento, especificando la información que se detallará en cada uno de ellos, de forma que se facilite al lector la ubicación de la información.

En el siguiente capítulo se ahonda en las tecnologías empleadas por el sistema de vídeo vigilancia con cámaras IP desarrollado en este Trabajo. Desde un punto de vista científico y técnico se expone las características de las cámaras de vídeo vigilancia analógicas frente a las IP. También se detallan las características de los protocolos, códec y formato empleados en la transmisión y grabación del vídeo de las cámaras.

El tercer capítulo detalla las aplicaciones *software* y los recursos *hardware* empleados en el desarrollo de la aplicación de vídeo vigilancia. Se profundiza en las características de cada uno, centrándose en su utilización en este Trabajo.

En el capítulo cuarto se exponen los requisitos funcionales y no funcionales del sistema de vídeo vigilancia IP en su conjunto, tanto en su parte *software* como en el equipo *hardware*.

A continuación, en el quinto capítulo se detalla el trabajo realizado en GMV para lograr la integración del nuevo modelo de cámaras. La finalidad del mismo es profundizar en el funcionamiento de la aplicación.

En el capítulo sexto se presenta un breve manual para el usuario que utiliza la aplicación de grabación de vídeos en el equipo.

El séptimo capítulo explica cada una de las pruebas realizadas al sistema, así como los resultados de las mismas. La finalidad del mismo es mostrar todo el proceso de detección de errores y resolución de los mismos hasta dar con una solución que cumpla con los objetivos planteados.

En el penúltimo capítulo, el octavo, se detalla la estimación del presupuesto económico para el desarrollo de la aplicación de grabación.

El noveno, y último capítulo pretende abrir las líneas futuras de mejora del sistema, proponiendo nuevas posibilidades que puedan implementarse próximamente tomando como pilar básico la labor realizada en este Trabajo. Además, se presentan las conclusiones extraídas de la realización del mismo.

Capítulo 2

Tecnologías base

2.1. Introducción

En este segundo capítulo se analiza tanto la evolución de las cámaras de vídeo vigilancia como los protocolos, la codificación y el formato de fichero utilizados en la grabación de los vídeos.

En una primera sección se va a explicar el funcionamiento de las cámaras analógicas en contraposición con el funcionamiento de las nuevas cámaras digitales IP, mostrando las ventajas y las desventajas de ambos tipos de cámara.

A continuación, se detallan los protocolos de transporte y de control utilizados para la transmisión de vídeo desde la cámara hasta el equipo de grabación.

Por último, se describen las principales características de la codificación y del formato de vídeo empleado para la grabación en el equipo destinado a este fin.

2.2. Tecnología de las cámaras de vídeo vigilancia

La vídeo vigilancia se remonta a los años 50 en los Estados Unidos y el Reino Unido [15], momento en el que se empleaba en bancos, tiendas y para el control del tráfico. En los años 70 se empieza a utilizar la vídeo vigilancia también en escuelas, hospitales y transporte público urbano, entre otros. Sin embargo, estos sistemas no se instalarán en los espacios públicos hasta los años 90 con el desarrollo de los sistemas de multiplexación digital [15].

Los sistemas CCTV (*Closed Circuit Television*) de vídeo vigilancia están compuestos, además de por cámaras y monitores como los sistemas CCTV tradicionales, por un dispositivo de almacenamiento de vídeo o DVR (*Digital Vídeo Recorder*) en los circuitos analógicos, mientras que en las redes IP se utiliza un NVR (*Network Vídeo Recorder*).

En las siguientes subsecciones se van a presentar las principales diferencias entre vídeo vigilancia analógica y vídeo vigilancia IP, así como los beneficios de migrar hacia los sistemas digitales.

2.2.1. Visión general del sistema

Los sensores que se utilizan en las cámaras de vídeo vigilancia analógicas son sensores CCD (*Charge Coupled Device*), mientras que las cámaras IP pueden utilizar tanto sensores CCD como CMOS (*Complementary Metal Oxide Semiconductor*) [16]. La señal analógica de ambos sensores debe digitalizarse para posteriormente ser procesada por el DSP (*Digital Signal Processor*). En este punto es donde empiezan las mayores diferencias entre ambos tipos de cámaras, tal y como se puede observar en la figura 1.

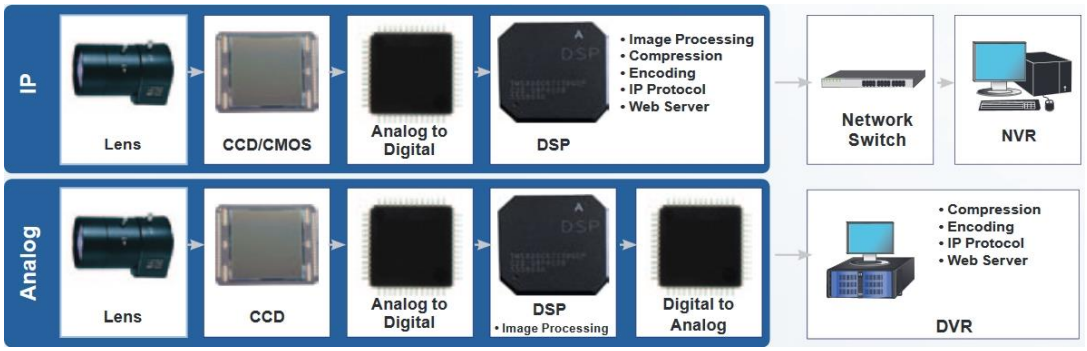

Figura 1. Sistema de vídeo vigilancia IP (arriba) y analógico (abajo) **[16]**.

En los sistemas IP, la imagen es comprimida y codificada en la propia cámara para ser transmitida por medio del protocolo IP. Esta imagen se almacena en el NVR que está conectado a la red IP. Algunos modelos de cámaras también incluyen la posibilidad de almacenar las imágenes en la propia cámara. Además, las cámaras IP incluyen un servidor web para poder acceder a ellas desde cualquier ordenador de la red o desde otro equipo con acceso a esta red [16].

Los sistemas analógicos funcionan de manera diferente, puesto que una vez procesada la imagen en el DSP, esta se convierte de nuevo a señal analógica y se

transmite a un monitor o al DVR. El DVR es el encargado de codificar y almacenar el vídeo, además de implementar el servidor web.

Por lo tanto, desde un punto de vista general y sin entrar en detalle, las principales diferencias son dónde se comprime el vídeo y qué componentes forman el sistema de vídeo vigilancia. Sin embargo, en los siguientes apartados se van a analizar las ventajas y desventajas de ambos sistemas en diferentes aspectos.

2.2.2. Calidad del vídeo

Las cámaras IP destacan en la captura de imágenes de alta definición, pero en condiciones de baja luminosidad y captura de movimiento se comportan peor que las cámaras analógicas. Esta situación puede observarse en la siguiente figura, donde se aprecia que la persona en movimiento tiene mejor definición en la captura de la cámara analógica:

Figura 2. Calidad del vídeo en una cámara IP (izquierda) y una analógica (derecha) **[16]**.

Además, debido a que la codificación en los sistemas de vídeo vigilancia IP se realiza en la propia cámara, se debe llegar a un equilibrio entre el códec, la tasa de fotogramas y la calidad, puesto que los recursos son limitados y la calidad de uno va en detrimento de los otros dos [16]. Esta codificación realizada en la propia cámara IP antes de la monitorización, implica que las imágenes tampoco se van a ver en su máxima calidad ni en tiempo real.

2.2.3. Transmisión de vídeo

Mientras que las cámaras analógicas utilizan cables coaxiales para la transmisión del vídeo, las cámaras IP pueden utilizar una infraestructura de red ya existente para implementar el sistema de vídeo vigilancia [16]. Además, las cámaras IP tienen la posibilidad de utilizar cables PoE (*Power over Ethernet*), los cuales permiten alimentar la cámara con el mismo cable que se utiliza para la transmisión de datos y vídeo.

Un inconveniente presente en el tráfico IP, que no sucede al utilizar cámaras analógicas, es que un fallo momentáneo en la red debido a la congestión, la limitación del ancho de banda, las tasas de bits variables, el balanceo de carga, posibles ficheros grandes, los virus o la latencia provoca una degradación del vídeo.

2.2.4. Fiabilidad, seguridad y coste

Las cámaras IP tienen menos fiabilidad que las analógicas debido a los fallos de la red IP que se han mencionado en el punto anterior. Esto también pone de manifiesto la necesidad de un buen mantenimiento en este tipo de sistemas de vídeo vigilancia IP. Los fallos que pueden tener lugar en las redes de cámaras analógicas son fallos individuales en alguno de los dispositivos hardware, por lo que en principio no provocan una degradación importante del sistema [16].

Dado que los flujos de vídeo IP se pueden encriptar, resultan difíciles de interceptar. En el caso de las redes de cámaras analógicas, se pueden considerar menos seguras porque cualquiera con acceso a la infraestructura cableada puede interceptar las señales. Sin embargo, a diferencia de las redes IP, el sistema de cámaras analógicas no está expuesto a ataques de hackers ni a virus, a excepción del DVR que está conectado a la red.

Lo presentado anteriormente pone de manifiesto la necesidad de realizar un buen mantenimiento en los sistemas de cámaras IP, además de garantizar la seguri-

dad y el aislamiento de la red con medios como cortafuegos y de dotarla de redundancia para aquellos sistemas que sean críticos. Es por esto que las soluciones de cámaras IP son más caras que las analógicas [16].

2.2.5. Sistemas inalámbricos

Una de las ventajas más evidentes de las cámaras IP es la posibilidad de integrar el sistema de vídeo vigilancia con redes inalámbricas, ya que en el mercado existen cámaras IP inalámbricas que por ejemplo funcionan mediante Wi-Fi.

Las cámaras analógicas que transmiten vídeo de manera inalámbrica lo hacen por radiofrecuencia, lo cual limita aproximadamente a una docena el número de cámaras que es posible utilizar en las bandas libres de radio [16].

2.2.6. Sistemas híbridos

Los sistemas híbridos consisten en una combinación de un sistema de vídeo vigilancia con cámaras analógicas y un sistema con cámaras IP, los cuales trabajan en paralelo. Una posible configuración podría ser la mostrada en la siguiente figura:

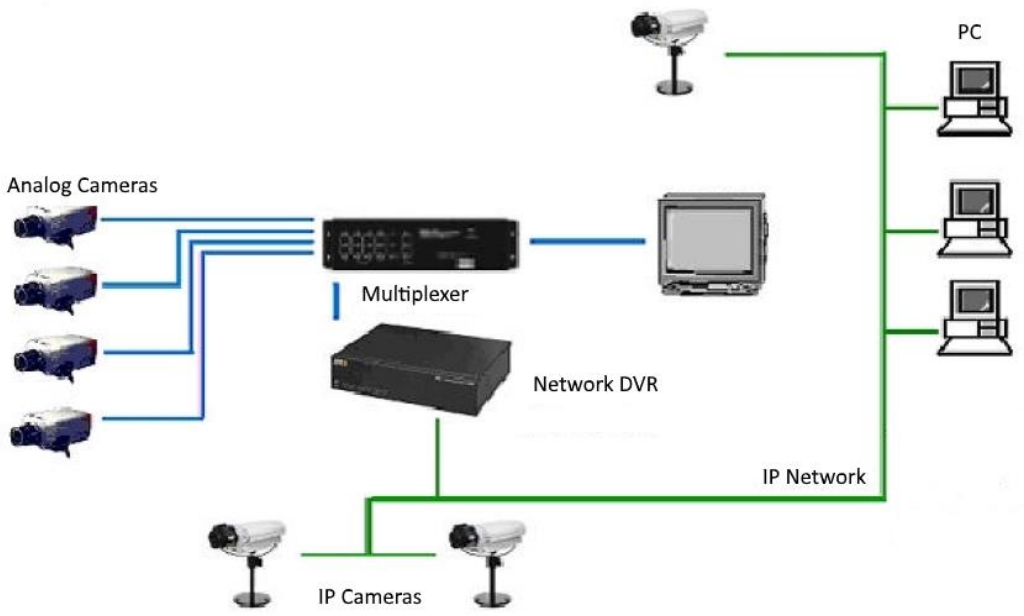

Figura 3. Sistema híbrido **[17]**.

En esta configuración se diferencian claramente dos partes. La primera sería la formada por las cámaras analógicas que se conectan a un multiplexador. Este multiplexador es el encargado de transmitir la señal al monitor para su visualización y al DVR para grabar el vídeo. Este DVR es al mismo tiempo un NVR, ya que además de codificar y almacenar el vídeo de las cámaras analógicas, también almacena el vídeo de las cámaras IP. La segunda parte estaría formada por la red IP, en la cual se encuentran las cámaras IP, el NVR y los equipos conectados a la red.

Por lo tanto, los sistemas de vídeo vigilancia no tienen por qué ser completamente analógicos o completamente IP, ya que es posible utilizar soluciones intermedias que combinen los beneficios de ambos sistemas, tal y como se acaba de ver. Sin embargo, estos sistemas híbridos no son habituales en los nuevos despliegues, sino que se suelen utilizar durante la transición paso a paso de un sistema analógico a un sistema de vídeo vigilancia IP [17].

2.3. Protocolos

La necesidad de comunicaciones multimedia surgió con el crecimiento de Internet, pero el tráfico de este tipo de comunicaciones plantea nuevos requisitos a la arquitectura de Internet y sus protocolos. Por este motivo se diseñó un conjunto de protocolos entre los que se encuentra RTP (*Real-Time Transmission Protocol*), RTCP (*Real-Time Transmission Control Protocol*) y RTSP (*Real-Time Streaming Protocol*).

El modelo de servicio *best effort* de IP ha demostrado ser una solución adecuada para el transporte de texto [18]. En este modelo, IP hace todo lo posible para entregar los paquetes pero no da ninguna garantía de que vayan a ser entregados. Por lo tanto, los paquetes se pueden perder o entregar desordenados y el retraso es impredecible. Este modo de funcionamiento permite que los *rourters* sean lo más simples posible y la arquitectura de la red sea fácilmente escalable.

El protocolo de transporte TCP (*Transmission Control Protocol*) permite aumentar la fiabilidad de los servicios extremo a extremo. De esta manera se solventa la falta de fiabilidad de IP, ya que permite retransmitir los paquetes perdidos, entrega los paquetes ordenados y mantiene controlada la congestión de la red. Por lo tanto, IP se implementa en todos los nodos y es simple y escalable, mientras que TCP se implementa sólo en los nodos de los usuarios finales y aporta fiabilidad a las aplicaciones [18].

TCP sobre IP es una buena solución, pero no es suficiente para transportar tráfico multimedia, ya que muchas aplicaciones son especialmente sensibles al retardo extremo a extremo y a las variaciones del retardo (*jitter*). En el caso de producirse pérdidas de paquetes, TCP retransmite los paquetes perdidos, pero no garantiza el retardo. Por este motivo, TCP no es adecuado para transportar el tráfico multimedia [18]. La solución es utilizar UDP (*User Datagram Protocol*) como capa de transporte sobre IP.

En cuanto a las aplicaciones multimedia, pueden dividirse en dos clases: aplicaciones interactivas en tiempo real y aplicaciones de *streaming* no interactivas. Las aplicaciones en tiempo real como VoIP (*Voice Over Internet Protocol*) y vídeo conferencia, precisan que toda la comunicación sea en tiempo real, por lo que los datos no se deben guardar demasiado tiempo en memoria para reducir el retardo de la red o el *jitter*. Las aplicaciones no interactivas pueden dividirse a su vez en dos clases: *streaming* de audio/vídeo almacenado en un servidor y *streaming* de audio/vídeo en directo. Ambas subclases son menos restrictivas que las aplicaciones interactivas en tiempo real, por lo que hacen uso del almacenamiento en memoria para reducir el *jitter* y el retardo.

A parte del retardo y *jitter* limitados, las aplicaciones multimedia precisan de ciertos servicios de transporte con características distintas de TCP y más funcionalidades que UDP. El protocolo diseñado para proporcionar estos servicios es RTP. Una de las cualidades de RTP es que permite al receptor reproducir los paquetes multimedia, guardados en memoria para reducir el *jitter*, en el orden correcto a

través de la relación de tiempos [18]. De la misma manera, las aplicaciones multimedia logran la sincronización utilizando RTP, por ejemplo para sincronizar los flujos de audio y vídeo.

En las siguientes subsecciones se van a presentar los protocolos RTP y RTSP, centrándose en los aspectos más importantes para la aplicación de vídeo vigilancia IP desarrollada en este Trabajo.

2.3.1. RTP

RTP proporciona funciones de transporte extremo a extremo adecuadas para las aplicaciones en tiempo real, como audio, vídeo o simulación de datos. RTP no se encarga de la reserva de recursos ni garantiza QoS (*Quality of Service*) para los servicios [19]. El estándar RTP define dos protocolos: RTP y RTCP. RTP se ocupa del intercambio de datos multimedia, mientras que RTCP se encarga de obtener información de control sobre la calidad de la transmisión en los receptores. Ambos protocolos están diseñados para ser independientes de las capas de transporte y de red subyacentes.

El vídeo y/o audio transportado por RTP se digitaliza utilizando un códec particular, que se presentará en los siguientes apartados. Estos bloques generados por el flujo de bits se encapsulan en paquetes RTP y, habitualmente, después en paquetes UDP e IP [18].

Cada paquete generado tiene una cabecera RTP en la que se incluye información especialmente útil para la reconstrucción de los datos multimedia. En esta cabecera se incluye el valor *Payload Type* para indicar qué tipo de codificación de audio o vídeo se ha utilizado en el transmisor, de manera que el receptor puede seleccionar el esquema de decodificación adecuado. También contiene información de los tiempos de los paquetes (*Timestamp*) y el número de secuencia, lo cual permite al receptor reconstruir la línea de tiempos generada en el transmisor para audio y vídeo [19].

2.3.2. RTSP

RTSP es un protocolo de la capa de aplicación, el cual sirve para configurar y controlar la transmisión de datos en tiempo real. Dicho de manera simple, RTSP se comporta como "un control remoto de red" para los servidores multimedia [20]. En concreto, el propósito de este protocolo es controlar sesiones múltiples de transmisión de datos, proporcionar un medio para seleccionar los canales de transmisión como UDP, UDP *multicast* y TCP, y proporcionar un medio para seleccionar mecanismos de transmisión basados en RTP. Trabaja entre los clientes y los servidores RTSP. Además, está diseñado para trabajar con protocolos de nivel inferior como RTP [18].

Es un protocolo bidireccional para peticiones y respuestas, que primero se encarga de establecer un contexto incluyendo los contenidos y después controla la transmisión de estos contenidos desde el emisor hasta el receptor. RTSP tiene tres partes fundamentales: establecimiento de la sesión, control de la transmisión de datos y un modelo de ampliación del sistema [20].

Los mensajes, peticiones y respuestas RTSP están basados en texto. Una petición RTSP comienza con una línea que identifica el método, el protocolo y la versión del recurso sobre el que actúa. Los recursos se identifican por medio de un URI (*Universal Resource Identifier*), las cuales habitualmente son URLs (*Universal Resource Locator*) e indican la ubicación del recurso. Las respuestas RTSP son similares, pero comienzan indicando el protocolo y la versión, seguidos por el código de estado.

Los servicios y operaciones que ofrece RTSP se utilizan por medio de llamadas a métodos. Los métodos indican qué se debe hacer sobre el recurso identificado con un URI en la petición. Estos métodos son: OPTIONS, DESCRIBE, ANNOUNCE, SETUP, PLAY, PAUSE, TEARDOWN, GET_PARAMETER, SET_PARAMETER, REDIRECT y RECORD. Sin embargo, no es necesario utili-

zar todos los métodos RTSP para que una transmisión sea completamente funcional [18]. A continuación se va a describir brevemente cada uno de ellos:

- OPTIONS: El cliente o el servidor informa al otro extremo de las opciones que acepta.
- DESCRIBE: El cliente recupera del servidor la descripción de un recurso identificado con el URI incluida en la petición.
- ANNOUNCE: Al enviarse desde el cliente al servidor, establece la descripción de un recurso en el servidor. Si se envía desde el servidor al cliente, actualiza la descripción de la sesión en tiempo real.
- SETUP: El cliente solicita al servidor asignar recursos para un flujo de datos e inicia una sesión RTSP.
- PLAY: El cliente pide al servidor que comience la transmisión de datos a través del flujo asignado con SETUP.
- PAUSE: El cliente detiene temporalmente el flujo de datos, pero sin liberar los recursos del servidor.
- TEARDOWN: El cliente solicita al servidor detener la transmisión de un flujo concreto y libera los recursos asociados.
- GET_PARAMETER: Obtiene el valor de un parámetro concreto del recurso especificado en el URI.
- SET_PARAMETER: Establece el valor de un parámetro concreto del recurso especificado en el URI.
- REDIRECT: El servidor informa a los clientes de que se deben conectar a otro servidor.
- RECORD: El cliente inicia la grabación de un conjunto de datos multimedia.

En los siguientes apartados se desarrollarán aquellos métodos especialmente importantes para la aplicación presentada en este Trabajo.

2.3.2.1. Método OPTIONS

Las peticiones OPTIONS pueden utilizarse en cualquier momento, ya que no modifican el estado de la sesión. La respuesta a un método OPTION es una lista de los métodos soportados por el agente RTSP. Además, se puede utilizar este método para mantener una sesión RTSP abierta [20].

2.3.2.2. Método DESCRIBE

El método DESCRIBE se utiliza para obtener la descripción de un recurso del servidor. Este recurso se identifica por medio del URI presente en la petición. La respuesta a esta petición debería contener toda la información de inicialización de los recursos a los que se refiere.

En un sistema RTSP es necesario inicializar los recursos. Sin embargo, las especificaciones de RTSP no obligan a que deba hacerse mediante el método DESCRIBE [20].

2.3.2.3. Método SETUP

La petición SETUP sobre un URI especifica el mecanismo de transporte a emplear para la transmisión multimedia [20]. Se puede utilizar para crear una sesión RTSP o para modificar los parámetros de una transmisión ya configurada.

En la respuesta SETUP 200 OK se deben incluir las propiedades multimedia en la cabecera.

2.3.2.4. Método PLAY

El método PLAY informa al servidor de que puede comenzar el envío de datos a través del mecanismo configurado anteriormente con SETUP. Se puede especificar el intervalo de tiempo del contenido multimedia que se quiere recibir. Si no se especifica ningún rango de tiempos en la petición PLAY, el servidor reproducirá desde el punto actual de pausa hasta el final del contenido multimedia [20]. El punto de pausa siempre es el instante actual en aquellos contenidos multimedia que son transmisiones en directo, ya que no se pueden pausar.

2.4. Codificación H.264

H.264, H.264/AVC, MPEG-4 parte 10 o MPEG-4-AVC son diferentes nombres para la norma que define un códec de vídeo de alta compresión [21], desarrollada por ITU-T *Vídeo Coding Experts Group* (VCEG) y ISO/IEC *Moving Picture Experts Group* (MPEG). Surge debido a la necesidad de mayor compresión en las imágenes en movimiento de aplicaciones como vídeoconferencia, almacenamiento multimedia, *streaming* por Internet y comunicaciones [22].

La codificación H.264 es capaz de reducir el tamaño de un archivo de vídeo digital más de un 80% en comparación con M-JPEG, y hasta un 50% comparándolo con el estándar MPEG-4 Parte 2 [23]. Esta reducción en tamaño implica que el ancho de banda y el espacio de almacenamiento requeridos para los vídeos es menor. O, visto de otra manera, para una tasa de bits determinada se logra mayor calidad de vídeo. Este códec es especialmente útil en los sistemas de vídeo vigilancia, ya que es donde se necesitan mayores velocidades y resoluciones.

Antes de entrar en los principales detalles que hacen de H.264 una codificación superior a las demás, se debe explicar el funcionamiento básico de la compresión de vídeo. La compresión de vídeo consiste en reducir y eliminar datos redundantes del vídeo para que el archivo pueda enviarse y almacenarse de manera eficiente [24]. Para ello, se aplica un algoritmo al vídeo original, creando un archivo

comprimido listo para ser enviado o guardado. Aplicando el algoritmo inverso, se obtiene un vídeo que contiene prácticamente el mismo contenido que el original. El tiempo empleado en comprimir, transmitir, descomprimir y mostrar el archivo es lo que se define como latencia. Este par de algoritmos que funcionan conjuntamente se denomina códec de vídeo, y son el codificador y el decodificador.

Al igual que otros estándares de vídeo, H.264 dispone de perfiles que determinan sus características algorítmicas. El perfil denominado "base" está destinado a aplicaciones con recursos informáticos limitados, por lo que habitualmente es el más adecuado para las cámaras IP [24]. Este perfil permite tener una baja latencia, siendo idóneo para la codificación en tiempo real.

2.4.1. Fotogramas

En función del perfil de H.264, el codificador puede emplear diferentes tipos de fotogramas: fotogramas I, fotogramas P y fotogramas B.

Un fotograma I o intrafotograma es una imagen codificable de manera independiente, sin hacer referencia a otras imágenes. La primera imagen de un vídeo siempre es de este tipo. Estos fotogramas sirven como puntos de inicio o puntos de resincronización en caso de fallo en la transmisión, por ello se les denomina también *keyframes*. El codificador inserta fotogramas I a intervalos regulares o a petición de los clientes. Este tipo de fotogramas tiene el inconveniente de consumir más bits, pero la ventaja de producir menos artefactos o defectos [24].

Un fotograma P o interfotograma predictivo hace referencia a partes de fotogramas I o P anteriores para codificarlo. Estos fotogramas requieren menos bits que los fotogramas I. Sin embargo, tienen una dependencia compleja con los fotogramas anteriores, lo cual provoca que sean más sensibles a los errores.

Un fotograma B o interfotograma bipredictivo hace referencia a fotogramas I o P tanto anteriores como posteriores.

En la siguiente figura se muestra un ejemplo de cómo podría ser una transmisión de vídeo codificada, en la que se utilicen los tres tipos de fotogramas:

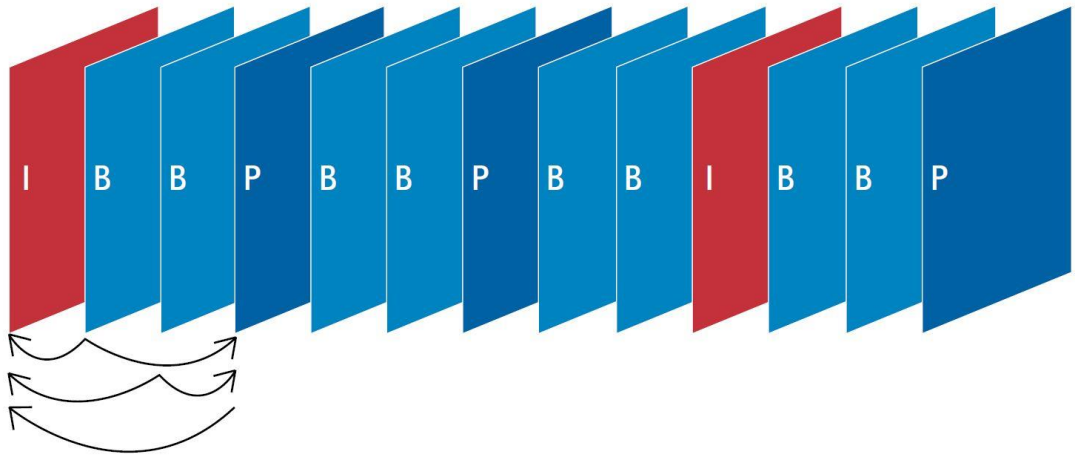

Figura 4. Secuencia de fotogramas **[24]**.

La decodificación de la transmisión de vídeo siempre debe comenzar por un fotograma I, para continuar fotograma a fotograma. Los fotogramas P y B deben decodificarse junto a sus correspondientes fotogramas de referencia.

Anteriormente se ha explicado que en las cámaras IP habitualmente se utiliza el perfil base de H.264. La latencia se reduce utilizando este perfil porque prescinde de los fotogramas B y utiliza únicamente fotogramas I y P [24].

2.4.2. Reducción de datos

La codificación diferencial, que se utiliza en H.264, permite reducir la cantidad de datos de un vídeo. Esta codificación compara un fotograma con otro fotograma anterior de referencia y codifica únicamente los píxeles que han cambiado. En la siguiente imagen se ve un ejemplo: La primera imagen sería un fotograma I y se codificaría completamente. Las siguientes dos imágenes son fotogramas P, en los cuales se codifican exclusivamente las partes en movimiento (diferencias) utilizando vectores de movimiento, que en este caso sería el hombre corriendo. Así, se reduce la cantidad de información transmitida y almacenada.

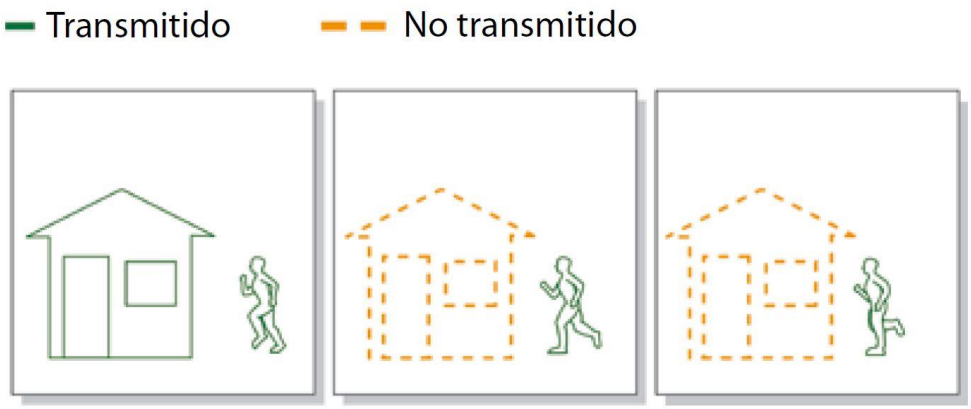

Figura 5. Codificación diferencial **[24]**.

La codificación se puede reducir más si en vez de pixeles se comparan áreas más grandes, ya que se codifican y decodifican bloques de pixeles o macrobloques. En caso de que sea un vídeo con mucho movimiento, la codificación diferencial no producirá una reducción significativa de los datos [24]. En ese caso se puede utilizar la compensación de movimiento basada en bloques, la cual tiene en cuenta que gran parte de la información de un fotograma ya está incluida en el fotograma anterior, aunque quizás en una posición diferente del mismo [24]. Con esta técnica, cuando se produce una coincidencia de un bloque en el fotograma de referencia, se codifica sólo la posición de ese bloque.

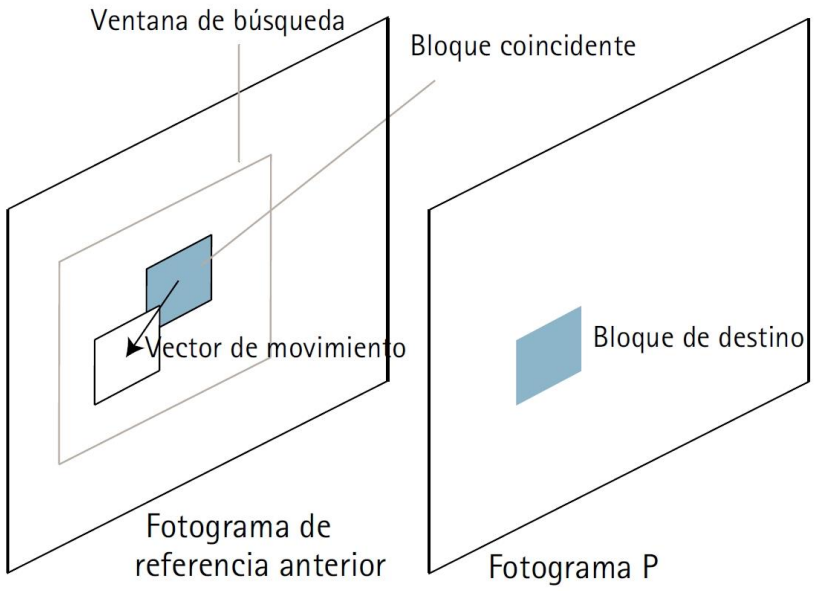

Figura 6. Compensación de movimiento **[24]**.

Otra función que incluye H.264 es la posibilidad de reducir los defectos o artefactos pixelados, que aparecen en las imágenes de vídeo muy comprimidas utilizando los estándares M-JPEG y MPEG. El filtro de eliminación de bloques en bucle es el encargado de suavizar los bordes de los bloques obteniendo una imagen casi perfecta [24].

2.5. Formato de fichero MP4

MP4 o MPEG-4 parte 14 es un formato de archivos o formato contenedor, que permite combinar audio, vídeo, subtítulos e imágenes estáticas en un único fichero. Este formato se utiliza para distribuir contenido que cumpla el estándar MPEG-4 [25], como podría ser H.264 para vídeo.

MP4 fue diseñado por ISO/IEC *Moving Picture Experts Group* para facilitar el intercambio de contenido [25], ya que el archivo es de tamaño reducido y cada uno de ellos contiene todos los datos necesarios para ser intercambiado. Es un formato compacto, jerárquico y estructurado.

Los datos multimedia y los metadatos están separados en este formato. Los datos multimedia son las muestras de vídeo o audio, mientras que los metadatos se utilizan para indicar el tipo de flujo, el *timestamp* o marca de tiempo de las muestras y la posición y longitud de las muestras, entre otras cosas.

Los archivos MP4 habitualmente son autocontenidos, es decir, contienen únicamente los datos multimedia que utilizan en su presentación, no referenciando datos de otros archivos [26].

El fichero MP4 se estructura como una secuencia de objetos llamados "cajas", algunos de los cuales a su vez contienen otros objetos [27]. En la siguiente imagen se aprecia esta estructura:

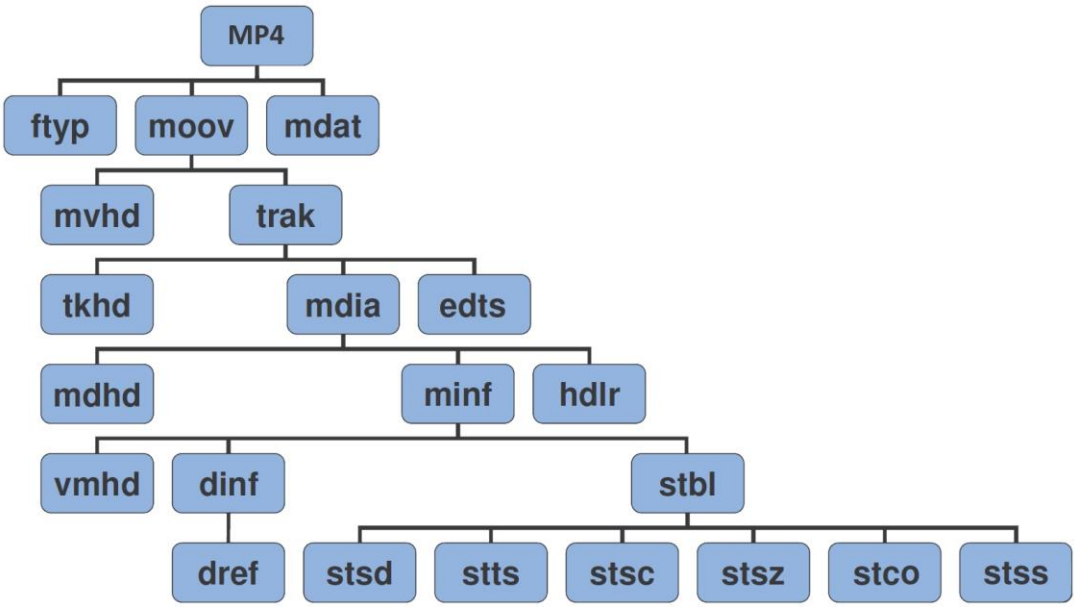

Figura 7. Fichero MP4.

Se pueden destacar una serie de objetos en la estructura del fichero MP4, que a su vez son los más importantes para la aplicación que se presenta en este Trabajo. Estos objetos o cajas son [26]:

- ftyp (*File Type*): Indica el tipo de fichero y la versión, para conocer la especificación que mejor se adapta al fichero. Esta caja debe situarse antes que las cajas de tamaño variable, como *Movie Box* y *Media Data Box*.

- mdat (*Media Data*): Contiene los datos multimedia, que en ficheros de vídeo serán fotogramas. Puede haber cero o más objetos *Media Data*.

- moov (*Movie Box*): Es el único contenedor de los metadatos. Habitualmente, este objeto se cierra al principio o al final del fichero, pero no es obligatorio.

- stsd (*Sample Description Box*): Es una tabla que da información detallada sobre el tipo de codificación utilizada y cualquier información de inicialización necesaria para esa codificación.

- stts (*Decoding Time to Sample Box*): Contiene una tabla compacta que permite indexar desde el tiempo de decodificación hasta el número de muestras. Cada entrada de la tabla proporciona el número de muestras consecutivas con la misma diferencia de tiempo, y la diferencia de tiempos de esas muestras. Así, sumando las diferencias de tiempos se puede obtener un mapeado de tiempo a muestra. Dicho de otra manera, el objeto stts indica cuánto tiempo se debe reproducir una muestra en unidades de la escala de tiempos [28].

- stsc (*Sample to Chunk Box*): Las muestras de los datos multimedia se agrupan en *chunks*, que es un conjunto de muestras contiguas. Los *chunks* pueden ser de diferentes tamaños, y sus muestras también. La tabla stsc indica cuántas muestras hay en cada *chunk* [28]. En el caso de la aplicación de vídeo vigilancia presentada en este Trabajo, se utiliza un único *chunk*.

- stsz (*Sample Size Boxes*): Contiene un contador de muestras y el tamaño en bytes de cada muestra.

- stco (*Chunk Offset Box*): Esta tabla indica el índice de cada *chunk* en el fichero.

- stss (*Sync Sample Box*): Esta tabla contiene un marcado de los puntos de acceso aleatorio del flujo. Es decir, indica qué fotogramas son *keyframes* o fotogramas I [28].

La estructura de los archivos MP4 generados con la aplicación de vídeo vigilancia de este Trabajo, observando exclusivamente el nivel superior de la jerarquía, es la siguiente:

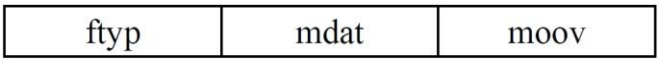

Figura 8. Estructura MP4.

El objeto moov debería generarse al principio del fichero [27], antes de almacenar los datos multimedia del objeto mdat. Sin embargo, en la solución de vídeo vigilancia, el objeto moov se genera al final. Esto es debido a que es una grabación de vídeo en tiempo real, por lo que primero se reciben y se guardan todos los datos multimedia del mdat (en función del tamaño de fichero configurado), para posteriormente generar el objeto moov correspondiente.

Capítulo 3

Hardware y *software* de la solución

3.1. *Software*

En el desarrollo de este Trabajo se han empleado diversas aplicaciones *software* para el estudio de las tramas y los ficheros y para el desarrollo de la aplicación.

- Eclipse: Entorno de desarrollo integrado de aplicaciones en diferentes lenguajes de programación. Entre sus funciones incluye: creación de proyectos, asistentes para la creación de proyectos, editor de texto con analizador sintáctico, compilador, navegador de carpetas y enlaces, generación de código, herramientas visuales de depuración, etc. Además, a través de plugins es posible añadirle más funcionalidades. En este trabajo se ha utilizado para programar la aplicación de grabación con las nuevas cámaras IP en lenguaje C/C++.

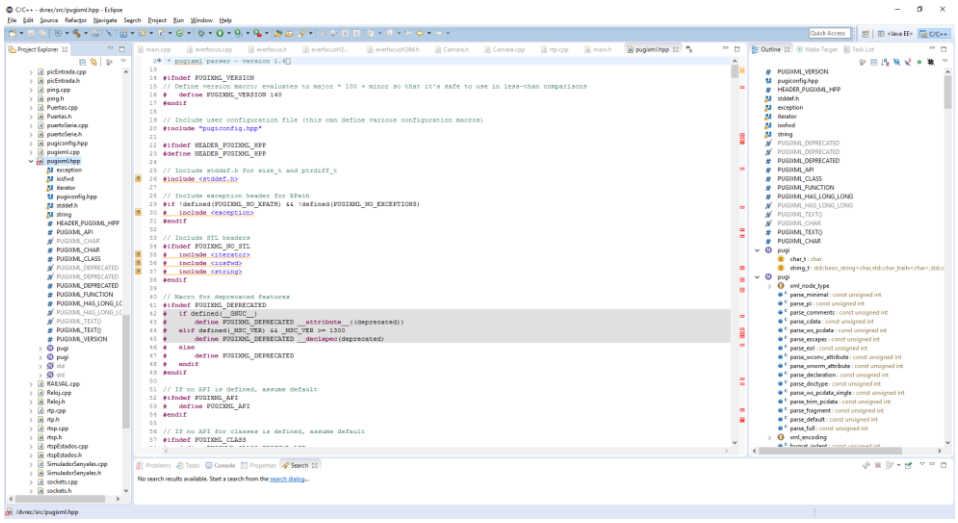

Figura 9. Pantalla principal del proyecto en Eclipse.

- MP4 Reader: Herramienta para analizar el contenido de ficheros de vídeo con formato MP4 y formatos derivados. Tal y como se aprecia en la figura 10, esta herramienta muestra la estructura jerárquica de cajas u objetos en la columna de la izquierda. En la ventana central muestra el contenido de cada caja de manera comprensible para el usuario, mientras que en la columna de la derecha muestra el contenido en hexadecimal.

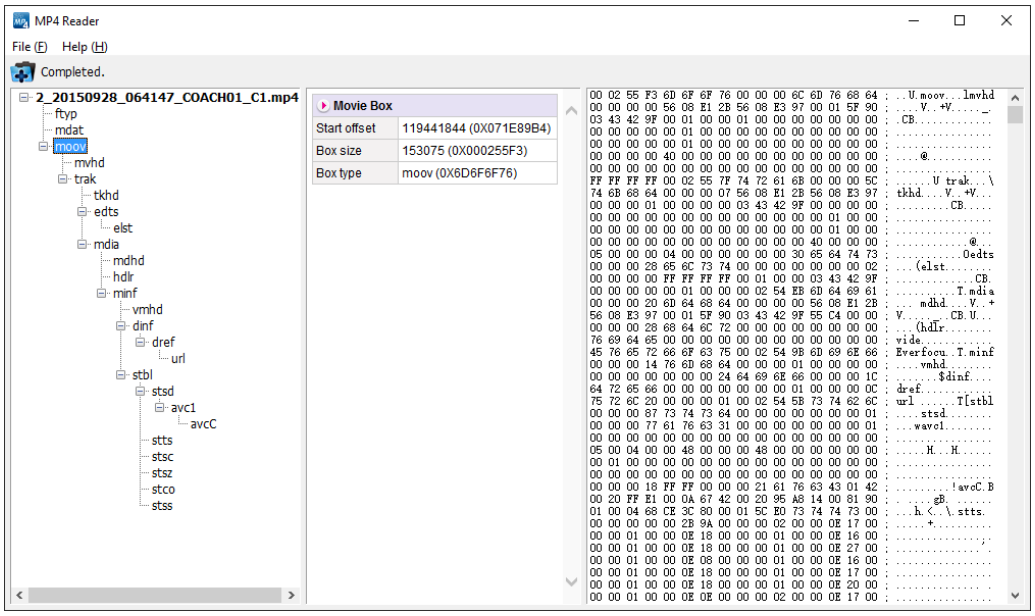

Figura 10. Fichero de vídeo analizado con MP4 Reader.

- Wireshark: Analizador de protocolos de red. Empleado en este trabajo para la captura de los paquetes generados en la transmisión de vídeo desde la cámara IP. El análisis de estos paquetes permite conocer el orden y tipo de los paquetes que contienen fotogramas de vídeo y la secuencia para configurar la transmisión de vídeo en la cámara.

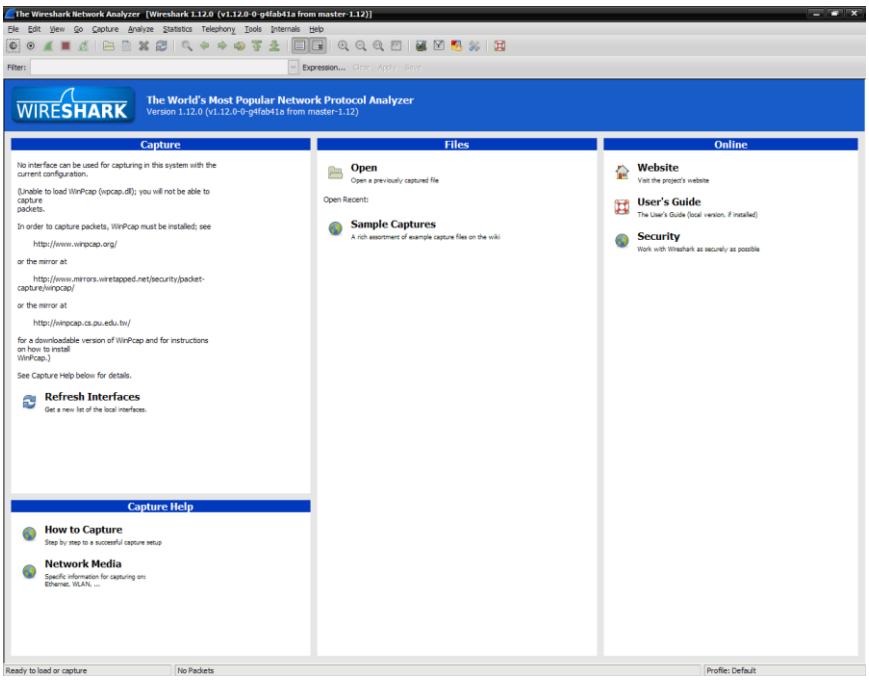

Figura 11. Pantalla principal del Wireshark.

- HxD: Editor hexadecimal. Dispone de funciones como editor de RAM, editor de disco, comparación de ficheros, generador de *checksum*, exportar datos a múltiples formatos, análisis básicos de datos, modo hexadecimal o texto, entre muchas otras. En este Trabajo se ha utilizado junto con el *software* MP4 Reader para estudiar el contenido de los ficheros de vídeo.

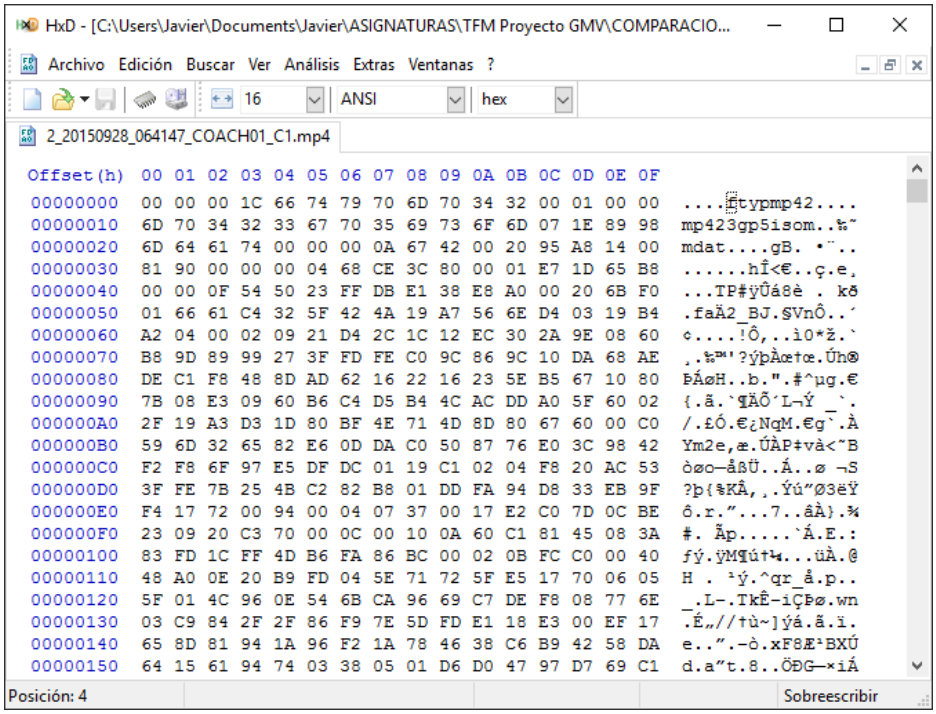

Figura 12. Fichero de vídeo analizado con HxD.

3.2. *Hardware*

Durante el desarrollo de la solución presentada en este Trabajo, se ha utilizado el siguiente dispositivo.

- Cámara Everfocus EMN2120 nevio HD Series [14]: Cámara IP encargada de la captura del vídeo y su transmisión al equipo de grabación. Sus características técnicas son las siguientes:

CÁMARA	
Megapíxeles	1.3 MP
Sensor	1/2.7" Full HD CMOS sensor
Lentes	2.8, 3.6, 6, 8 mm
Ilumminación	Color: 0.5 Lux Blanco y negro: 0.1 Lux
Tiempo de obturación	[4, 1/15000] seg
Ángulos	Pan: [-25°, +25°] Tilt: [0°, +90°] Rotación: [-90°, +90°]
VÍDEO	
Compresión	H.264 / MJPEG
Streaming de vídeo	2 streams simultáneos 5 conexiones unicast simultáneas
Tipos de resolución	PAL/NTSC
Resolución	Stream 1 (H264/MJPEG): 1280 x 1024, 1024 x 768, 720 x 576, 704 x 576, 640 x 480 Stream 2 (H264/MJPEG): 720 x 576, 704 x 576, 640 x 480, 352 x 288, 320 x 240
FPS	[1, 25] fps
Frecuencia	50 ó 60 Hz anti-fliker
SNR	> 50 dB
Privacy Mask	Hasta cuatro zonas
Region of Interest	Hasta 8 regiones
Calidad	VBR (*Variable Bit Rate, Fixed Quality*) CBR (*Constant Bit Rate*): [256 Kbps, 6 Mbps]

Configuración de imagen	Brillo, Saturación, Nitidez, Día o Noche, Control de exposición *Exposure control* *Flickerless*: [Off, 50Hz, 60Hz] AGC (*Auto Gain Control*) AWB (*Auto White Balance*) WDR (*Wide Dynamic Range*) 3DNR (*3D Noise Reduction*) DNR (*Digital Noise Reduction*) OSD (*On Screen Display*)
AUDIO	
Compresión	G.711
RED	
Interfaz	10BASE-T / 100BASE-TX
Protocolos y servicios	ARP, Bonjour, DDNS, DHCP, DNS, FTP, HTTP, HTTPS, ICMP, IGMP, NTP, PPPoE, RTCP, RTP, RTSP, SMTP, SNMP, TCP / IP, ONVIF, PSIA, UDP, UPnP
Seguridad	Contraseña Filtrado de IPs Encriptación HTTPS SSL
FUNCIONALIDADES	
Notificación	Email Subida de vídeo a FTP Grabación de vídeo en PC o SD CMS server HTTP event server
Alarmas	Manual Alarmas por horario Detección de movimiento Detección de tamper
Configuración inicial	Comandos CGI Fichero de parámetros binario

GENERAL	
Alimentación	10-36 Vdc, PoE (IEE 802.3af)
Consumo máx potencia	10-36 Vdc 7.5W / PoE 4.8W
Resistencia climática	IP67
Resistencia a vándalos	IK10-rated metal housing
Humedad relativa	[20%, 80%] sin condensaciones
Temperatura de operación	12 Vdc: [-40°C, 55°C] PoE: [-20°C, 55°C]
Conectores	DC Jack, Line in, RJ-45
Dimensiones (WxHxD)	112 x 46.5 x 130 mm
Peso	300 g
Varios	Micrófono integrado Soporte para tarjeta SD Monitorización en tiempo real desde dispositivos móviles

Tabla 1. Especificaciones técnicas de la cámara Everfocus **[14]**.

En la siguiente figura se muestra la cámara Everfocus utilizada durante el desarrollo del Trabajo:

Figura 13. Cámara Everfocus EMN2120 nevio HD Series **[14]**.

Capítulo 4

Análisis de la aplicación

4.1. Introducción

En este Trabajo se pretende integrar un nuevo modelo de cámaras en la solución de grabación existente. La solución previa funciona correctamente, pero se busca que sea compatible con otro modelo de cámara para reducir su coste y, de esta manera, poder llegar a un mercado más amplio.

La solución de vídeo vigilancia se emplea principalmente en tranvías y trenes, debido a que se ha desarrollado en el departamento ferroviario. Sin embargo, en otros departamentos utilizan sistemas similares, por ejemplo para autobuses.

Inicialmente, la integración de las nuevas cámaras se hará exclusivamente en un sistema de grabación de vídeo. El esquema de conexiones y elementos que conforman este sistema se muestra en la siguiente figura:

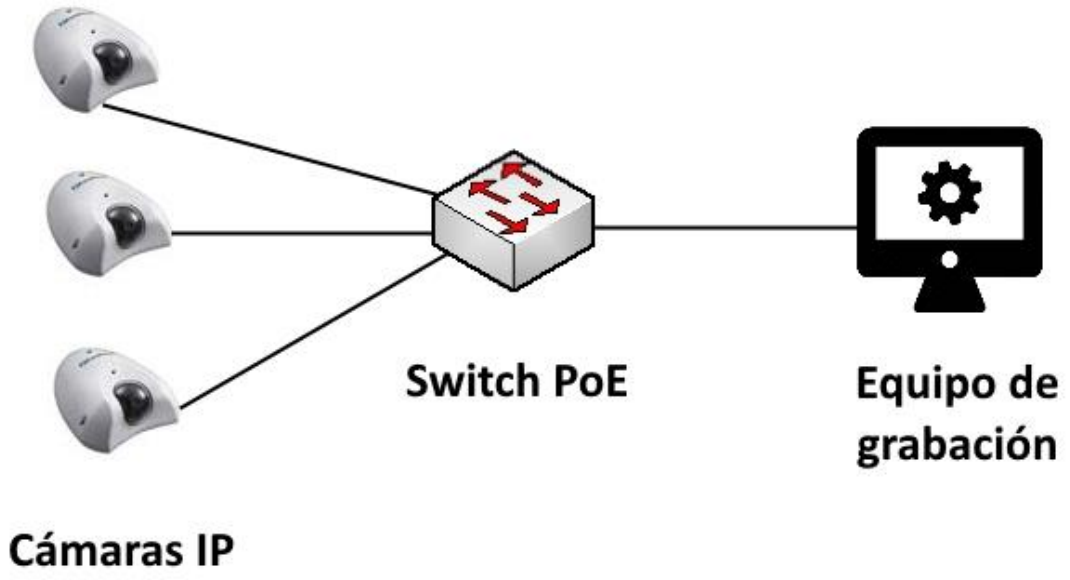

Figura 14. Sistema de grabación de vídeo.

El equipo de grabación digital es una unidad desarrollada por GMV, que tiene las siguientes características técnicas:

COMPONENTES	
Memoria RAM	1GB (expandible)
Memoria Flash	1 GB para el sistema operativo y las aplicaciones (expandible)
Dispositivo de almacenamiento de vídeo	SSD (*Solid State Disk*) o HD (*Hard Disk*) Capacidad para dos SSD/HD Actualmente se está utilizando un SSD SATA de 512 GB
Ventiladores	No
GENERAL	
Dimensiones (WxHxD)	271 x 88 x 221 mm
Peso	2.9 Kg
Alimentación	110 Vdc
Consumo más potencia	30 W
Sistema operativo	Linux optimizado para uso embebido
Interfaces	USB 2.0, RS-232, RS485, VGA opcional, soporte HMI opcional
RED	
Interfaz Ethernet	100 Mbps con conector M12
FUNCIONALIDADES	
Grabación de vídeo digital	
Compatibilidad con cámaras IP	
Gestión independiente de cámaras	
Acceso a las cámaras de a bordo en tiempo real utilizando HMI (opcional)	
Descarga remota de vídeo mediante WiFi (opcional)	
Conectividad GPS	
Transmisión de vídeo en tiempo real al Centro de Control (opcional)	

Tabla 2. Especificaciones técnicas del equipo de grabación **[29]**.

Tomando como ejemplo el sistema de vídeo vigilancia en un tranvía, en la figura 15 se ve la disposición de las cámaras IP en el mismo. Un tranvía puede componerse de dos a nueve coches, teniendo cada coche de una a cuatro cámaras de vídeo vigilancia. Como se puede observar, en el ejemplo mostrado se utiliza un total de cinco cámaras para cada tranvía. En cada puerta se sitúa una cámara para controlar el tránsito de los pasajeros y otra cámara se ubica en la cabina del conductor.

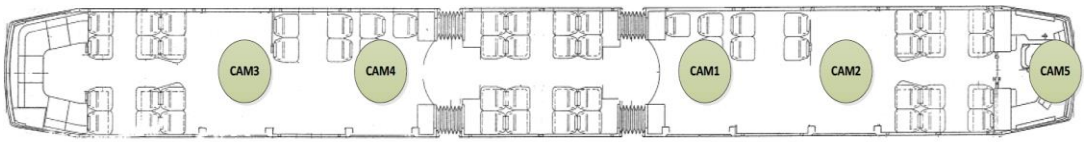

Figura 15. Ubicación de las cámaras en un tranvía **[29]**.

A pesar de que en este Trabajo sólo se presenta el sistema de grabación de vídeos, este se puede integrar con otros sistemas como:

- Posicionamiento del vehículo mediante GPS.
- Megafonía para la transmisión de audios.
- Intercomunicadores para la comunicación con zonas concretas de los vagones.
- Gestión de alarmas, avisos o fallos.
- Visionado de las cámaras en tiempo real desde el propio tren o tranvía.
- Acceso remoto a las cámaras situadas en el vehículo.

De esta manera se obtiene un sistema más complejo, pero con más funcionalidades. Un ejemplo sería el sistema SIV (Sistema de Información al Viajero). Este sistema ubica un terminal táctil en la cabina del maquinista, el cual permite realizar las siguientes acciones [29]:

- Cargar itinerarios automáticos que generen avisos sonoros y visuales de manera autónoma.
- Enviar textos predefinidos o libres a los indicadores interiores asociados a pasaje y a los indicadores exteriores visibles desde los andenes.
- Comunicarse por megafonía con los pasajeros, con los andenes o con ambos. Y recibir comunicaciones de los intercomunicadores instalados en cada puerta, si se selecciona la configuración correspondiente.
- Visualizar todas las cámaras del tranvía de manera simultánea o visualizar en pantalla completa una en concreto.
- Visualizar información de diagnosis de cada subsistema para informar en caso de fallo.
- Visualizar avisos si se activa un intercomunicador o si se ha solicitado parada.

4.2. Especificación de requisitos

Se pueden dividir los requisitos en dos grupos: funcionales y no funcionales. Los requisitos funcionales definen las acciones que ha de ser capaz de llevar a cabo la aplicación, de forma que la grabación de vídeo sea correcta. Mientras que los requisitos no funcionales son aquellos que no forman parte de la funcionalidad principal de la aplicación (fiabilidad, disponibilidad, costo, etc).

4.2.1. Requisitos funcionales

Se deben garantizar los siguientes requisitos de acuerdo a las características del sistema de grabación de vídeos con el que se desea integrar:

a) La aplicación ha de ser capaz de grabar el vídeo emitido por la cámara IP en ficheros de vídeo de longitud configurable.

b) La grabación de los vídeos ha de realizarse sin que se produzcan pérdidas de imágenes entre ficheros de vídeo consecutivos. Es decir, los tiempos de ficheros consecutivos deben estar perfectamente sincronizados.

c) El formato de los ficheros de vídeo que la aplicación será capaz de crear deben ser MP4.

d) El nombre de los ficheros de vídeo debe seguir el siguiente formato: [ID_tren] _ [Fecha_creación] _ [Hora_creación] _ [Vagón] _ [Número]. Además, los vídeos deben guardarse en una carpeta con la fecha en la que fueron creados.

e) La hora de las cámaras debe estar correctamente configurada y debe sincronizarse utilizando un servidor NTP *(Network Time Protocol*).

f) En las grabaciones de vídeo debe mostrarse la fecha y la hora de cada instante, superpuestas en una esquina de la imagen.

g) La configuración de la cámara IP debe ser modificable por código y se configurará al establecer la conexión, antes de comenzar la transmisión de vídeo.

h) Los vídeos grabados deben tener una serie de características configurables, para lo cual la transmisión de vídeo ha de ser configurable. Los parámetros a configurar son los siguientes:

    - Tasa de fotogramas por segundo. Habitualmente se configurarán 25 fps.
    - Resolución: altura y anchura.
    - Exploración o escaneo de vídeo: progresivo o entrelazado.
    - Calidad: VBR o CBR.

4.2.2. Requisitos no funcionales

Los requisitos no funcionales son los siguientes:

a) El nuevo modelo de cámaras debe alimentarse a través de PoE desde el propio *switch* instalado en el vehículo.

b) El coste de la solución utilizando el modelo de cámaras Everfocus, debe ser inferior al coste de la solución existente que empleaba cámaras Moxa.

c) Las cámaras deben poseer certificación de protección antivandálica y protección frente a agentes ambientales.

Capítulo 5

Diseño de la aplicación

5.1. Introducción

En este capítulo se presentará la integración del nuevo modelo de cámaras Everfocus con la solución de grabación existente. El desarrollo de esta integración se hizo sobre un entorno simplificado, en el cual se utilizó únicamente una cámara Everfocus conectada a un *switch* y un ordenador de sobremesa conectado también al *switch*. Además, como el *switch* no era PoE, la cámara IP se alimentó externamente a 24 Vdc. En la siguiente imagen se observa el esquema de conexiones, elementos y direcciones IP utilizados para el desarrollo:

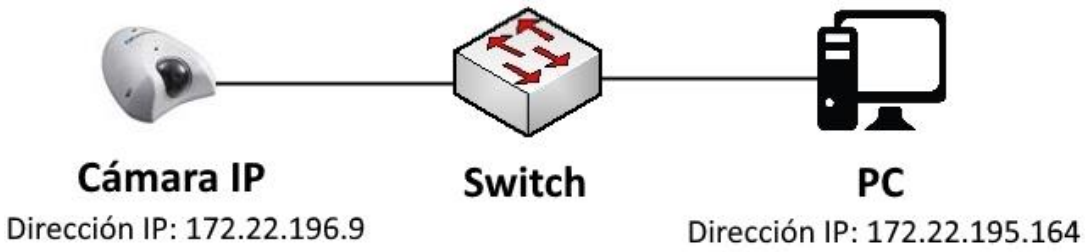

Figura 16. Sistema de desarrollo.

El desarrollo de la aplicación comenzó estudiando las tramas capturadas con Wireshark al visualizar la transmisión de vídeo de la cámara IP. Así, se pudo estudiar la secuencia RTSP intercambiada para la conexión, de manera que pudiese ser replicada por la aplicación.

A continuación, se repasó la documentación de la cámara para conocer el formato que debían tener los comandos de configuración. Y finalmente, se desarrolló el código necesario para que la solución existente pudiese grabar vídeo del nuevo modelo de cámaras IP.

5.2. Comandos CGI

La configuración inicial de la cámara Everfocus debe hacerse a través de comandos CGI (*Common Gateway Interface*). Estos comandos permiten interactuar

con la cámara, siendo posible controlar las mismas opciones que a través de la interfaz web. Entre todos los comandos disponibles se utilizan los siguientes:

- Activar el OSD (*On Screen Display*) para mostrar en pantalla el nombre de la cámara junto con la fecha y la hora de la grabación.

- Invertir la imagen en caso de que la cámara no está fijada en el techo.

- Configurar completamente el canal de vídeo seleccionado, especificando parámetros como canal, códec de vídeo, tipo de escaneo, resolución, etc.

- Configurar el DST (*Daylight Saving Time*) dependiendo de si se está en horario de verano o no.

- Ajustar la hora manualmente o de manera automática por NTP. En caso de ajustarse por NTP, se debe definir cuál será el servidor NTP.

- Opcionalmente se puede obtener información sobre el dispositivo, en caso de que sea necesario conocer parámetros como el identificador del dispositivo, el número de serie, la dirección MAC, la versión del *firmware* o la versión del *hardware*, entre otros.

Los comandos enviados a la cámara tienen la estructura mostrada en el ejemplo de la figura 17. Como se puede observar, primero se pone el método (GET, PUT, POST o DELETE), seguido del URI que identifica el recurso sobre el que se desea actuar y, detrás de este, se indica la versión de HTTP (*Hypertext Transfer Protocol*). Después se pone el nombre del agente usuario, la dirección IP de la cámara y se indica que se acepta todo tipo de contenido como respuesta. A continuación, en aquellas peticiones en las que se deban especificar parámetros, se indica el tipo de contenido incluido en la petición (en este modelo de cámaras es tipo XML), la longitud del mismo y, por último, se introducen los parámetros necesarios en formato de XML.

```
PUT /PSIA/Custom/EverFocus/ HTTP/1.1 User-Agent: gmv
Host: 172.22.196.9 Accept: */* Content-type: text/xml
Content-Length: 141

<?xml version="1.0" encoding="utf-8" ?>

<EverFocusCamSetting version="1.0" xmlns="urn:psialliance-
org">

    <Flip>true</Flip>

</EverFocusCamSetting>
```

Figura 17. Ejemplo de petición enviada a la cámara.

La respuesta de la cámara a estos comandos también tiene formato de XML. Para el procesamiento de las respuestas, se utiliza la librería pugiXML [30]. Es una librería simple y rápida para C++, la cual permite procesar XML. También soporta XPath, lo que permite buscar y seleccionar campos de la respuesta teniendo en cuenta la estructura jerárquica del XML.

Para saber si los comandos de modificación han tenido éxito, se analiza la respuesta de la cámara. Esta respuesta tiene la siguiente estructura:

```
<ResponseStatus>
    <requestURL>requestURL</requestURL>
    <statusCode>statusCode</statusCode>
    <statusString>statusString</statusString>
    <ID>0</ID>
</ResponseStatus>
```

Figura 18. Respuesta XML a los comandos de modificación **[31]**.

En concreto, basta con analizar el campo "*statusCode*" de la respuesta. El campo "*statusString*" presenta de manera comprensible el significado de la respuesta. En la tabla 3 se pueden observar los posibles contenidos de ambos campos.

statusCode	*statusString*
1	OK
2	Device Busy
3	Device Error
4	Invalid Operation
5	Invalid XML Format
6	Invalid XML Content
7	Reboot Required

Tabla 3. Campos *statusCode* y *statusString* de la respuesta **[31]**.

En los siguientes apartados se va a explicar el formato de los comandos CGI utilizados en la aplicación, así como los parámetros de los que constan.

5.2.1. Activación del OSD

El OSD se debe activar para que el nombre, la fecha y la hora de la cámara que está grabando, se muestren superpuestos en la imagen.

El comando emplea el método PUT para modificar un recurso del servidor, el cual tiene la URI "*/PSIA/Custom/EverFocus*". El contenido XML de la petición, donde se indican los parámetros a modificar y sus valores, es el siguiente:

```
<EverFocusSpecial>
      <EverFocusTimeStampSetting>
            <MachineName>true</MachineName>
            <DateTime>true</DateTime>
            <FrameRate>false</FrameRate>
            <ZoomRatio>false</ZoomRatio>
            <Position>UL</Position>
            <DateFormat>yyyy/m/d</DateFormat>
            <TimeFormat>24HR</TimeFormat>
      </EverFocusTimeStampSetting>
</EverFocusSpecial>
```

Figura 19. Petición de activación del OSD.

Este contenido XML indica que se debe mostrar únicamente el nombre de la cámara y la fecha y hora. La posición de estos elementos será la esquina superior izquierda, lo cual se indica con "*UL*" (*Up Left*). Por último, se especifica el formato de la fecha y de la hora.

5.2.2. Inversión de la imagen

La cámara está diseñada con la idea de ser fijada en el techo. En caso de que no se coloque en el techo, la imagen debe ser invertida para que se visualice correctamente.

Al igual que el comando anterior, se emplea el método PUT. El URI del recurso a modificar es "*/PSIA/Custom/EverFocus/CamSettingPlus*". En este caso, el contenido XML de la petición es bastante reducido, ya que es suficiente con indicar la activación del *flip* o inversión:

```
<EverFocusCamSetting>
      <Flip>true</Flip>
</EverFocusCamSetting>
```

Figura 20. Petición de inversión de la imagen.

5.2.3. Configuración del canal de vídeo

El vídeo debe configurarse antes de comenzar la transmisión desde la cámara. Para ello se utiliza el método PUT para modificar el recurso con URI "*/PSIA/Streaming/Channels/0*". El modelo de cámara Everfocus utilizado en este Trabajo tiene dos canales de vídeo, pero en la aplicación se utiliza únicamente el primer canal, tal y como aparece indicado en el URI.

En la configuración del canal de vídeo, también debe establecerse el protocolo de control a utilizar, el tipo de transmisión *unicast* o *multicast* y la seguridad. En esta aplicación, la transmisión es *unicast* y no se utiliza seguridad, ya que el sistema consiste en una red cableada cerrada.

El contenido XML de la petición es el mostrado a continuación:

```
<StreamingChannel>
      <id>0</id>
      <channelName>Stream1</channelName>
      <enabled>1</enabled>
      <Transport>
            <ControlProtocolList>
               <ControlProtocol>
                  <streamingTransport>RTSP</streamingTransport>
               </ControlProtocol>
            </ControlProtocolList>
            <Unicast>
                  <enabled>1</enabled>
                  <rtpTransportType>RTP/UDP</rtpTransportType>
            </Unicast>
            <Multicast>
                  <enabled>0</enabled>
            </Multicast>
            <Security>
                  <enabled>0</enabled>
            </Security>
      </Transport>
      <Vídeo>
        <enabled>1</enabled>
        <vídeoInputChannelID>0</vídeoInputChannelID>
        <vídeoCodecType>H.264</vídeoCodecType>
        <vídeoScanType>progressive</vídeoScanType>
        <vídeoResolutionWidth>1280</vídeoResolutionWidth>
        <vídeoResolutionHeight>1024</vídeoResolutionHeight>
        <vídeoQualityControlType>CBR</vídeoQualityControlType>
        <constantBitRate>1536</constantBitRate>
        <maxFrameRate>2500</maxFrameRate>
        <rotationDegree>0</rotationDegree>
        <mirrorEnabled>false</mirrorEnabled>
      </Vídeo>
</StreamingChannel>
```

Figura 21. Petición de configuración del canal de vídeo.

En esta configuración se empieza seleccionando el primer canal de vídeo con nombre "*Stream1*". A continuación, se debe elegir entre RTSP o HTTP como protocolo de control [31]. En caso de elegir transmisión *unicast*, se deberá seleccionar UDP o TCP como protocolo de transporte [31]. Por último, se configura el vídeo propiamente dicho:

- Canal de vídeo: Se selecciona otra vez el primer canal, ya que es el que tiene mejor resolución.
- Tipo de códec: Las opciones son MPEG4, MJPEG, 3GP, H.264 o MPNG. En esta aplicación se utiliza H.264.
- Tipo de escaneo: Dispone de escaneo progresivo o entrelazado. En este caso se selecciona progresivo.
- Resolución: Se selecciona la resolución indicando primero la anchura y después la altura del vídeo. En este ejemplo se establece la máxima resolución (1280x1024).
- Control de la calidad: Puede elegirse entre CBR (*Constant Bit Rate*) o VBR (*Variable Bit Rate*). En la aplicación se utiliza CBR y, a continuación, se establece una tasa de bits de 1536 Kbps.
- Tasa de fotogramas por segundo: Se selecciona la tasa de fotogramas deseada, multiplicada por 100. En este caso se selecciona la máxima, que tiene un valor de 25 fps.
- Rotación: Permite establecer los grados que estará rotada la imagen.
- Efecto espejo: Permite aplicar el efecto espejo al vídeo de la cámara.

5.2.4. Configuración del DST

El horario de verano debe activarse dependiendo de la época del año, de manera que se modificará automáticamente este comando en función de la fecha. El horario de verano comienza en abril y termina en noviembre.

Al igual que en los comandos anteriores, se emplea el método PUT. El recurso a modificar tiene URI "*/PSIA/Custom/EverFocus/Time*". En el contenido XML se establece el booleano que indica si está activado el DST o no. Además, es obligatorio indicar el intervalo de sincronización NTP, siendo el mínimo una hora.

```
<EverFocusTime>
  <NtpSyncInterval>1</NtpSyncInterval>
  <DaylightSavingEnable><!--boolean--></DaylightSavingEnable>
</EverFocusTime>
```

Figura 22. Petición de configuración del DST.

5.2.5. Ajuste de la hora

La hora de la cámara Everfocus se puede ajustar de manera manual, es decir, estableciendo la misma hora que tiene el equipo en el momento de mandar la petición. Otra posibilidad es ajustar la hora automáticamente cada cierto tiempo, utilizando un servidor NTP.

5.2.5.1. Ajuste manual de la hora

Para ajustar la hora manualmente en base a la hora del equipo que envía el comando, se utiliza el método PUT sobre el URI "*/PSIA/System/Time*".

El contenido del XML es como el mostrado en la figura 23. Se debe indicar modo manual, la fecha y hora como una cadena en formato ISO 8601 y la zona horaria.

```
<Time>
      <timeMode>manual</timeMode>
      <localTime><!--xs:datetime--></localTime>
      <timeZone>UTC-2:00</timeZone>
</Time>
```

Figura 23. Petición de ajuste de hora manual.

5.2.5.2. Ajuste automático de la hora

El ajuste de hora automático utilizando un servidor NTP se realiza enviando dos peticiones. Ambas peticiones utilizan el método PUT.

La primera petición es igual al comando del ajuste manual, con la salvedad de que se indica modo NTP y no es necesario especificar el campo de fecha y hora, tal y como se ve en la siguiente figura:

```
<Time>
      <timeMode>NTP</timeMode>
      <timeZone>UTC-2:00</timeZone>
</Time>
```

Figura 24. Primera petición de ajuste de hora automático.

La segunda petición se encarga de definir el servidor NTP con el cual la cámara sincronizará la hora. Esta petición modifica el recurso con URI "*/PSIA/System/time/NTPServers*". En el contenido XML del comando se pueden especificar varios servidores NTP, pero en este caso se define únicamente un servidor al que se le da un identificador. Para el formato de la dirección se utiliza una dirección IP, aunque se podría especificar con un nombre de *host*. En el último campo se debe indicar la dirección IP del servidor NTP. En la figura 25 se muestra este contenido XML.

```
<NTPServerList>
   <NTPServer>
      <id>0</id>
      <addressingFormatType>ipaddress</addressingFormatType>
      <ipAddress>172.22.195.167</ipAddress>
   </NTPServer>
</NTPServerList>
```

Figura 25. Segunda petición de ajuste de hora automático.

5.2.6. Obtención de información de la cámara

En caso de ser necesario se puede obtener la información de la cámara. Para ello se utiliza el método GET sobre el recurso con URI "*/PSIA/System/DeviceInfo*". El comando utilizado en este caso es más sencillo, ya que no se debe añadir ningún contenido XML.

La respuesta de la cámara IP a este comando tiene formato XML. En concreto, la respuesta tiene la siguiente estructura:

```
<DeviceInfo>
     <deviceName><!--xs:string--></deviceName>
     <deviceID><!--xs:string--></deviceID>
     <deviceDescription><!--xs:string--></deviceDescription>
     <systemContact><!--xs:string--></systemContact>
     <model><!--xs:string--></model>
     <serialNumber><!--xs:string--></serialNumber>
     <macAddress><!--xs:string--></macAddress>
     <firmwareVersion><!--xs:string--></firmwareVersion>
     <firmwareReleasedDate><!--xs:string-->
          </firmwareReleasedDate>
     <logicVersion><!--xs:string--></logicVersion>
     <logicReleasedDate><!--xs:string--> </logicReleasedDate>
     <bootVersion><!--xs:string--></bootVersion>
     <bootReleasedDate><!--xs:string--></bootReleasedDate>
     <rescueVersion><!--xs:string--></rescueVersion>
```

```
      <rescueReleasedDate><!--xs:string-->
            </rescueReleasedDate>
      <hardwareVersion><!--xs:string--></hardwareVersion>
      <systemObjectID><!--xs:string--></systemObjectID>
</DeviceInfo>
```

Figura 26. Respuesta a la petición de obtención de información.

Como puede observarse, devuelve los siguientes campos que aportan información sobre el dispositivo:

- Nombre.
- Identificador.
- Descripción del dispositivo.
- Contacto del sistema.
- Modelo.
- Número de serie.
- Dirección MAC.
- Versión del *firmware* y su fecha de publicación.
- Versión de la lógica y su fecha de publicación.
- Versión de arranque y su fecha de publicación.
- Versión de recuperación y su fecha de publicación.
- Versión del *hardware*.
- Identificador del objeto de sistema.

5.3. Aplicación de grabación

En este apartado se explica el funcionamiento de la aplicación de grabación de vídeos. Sin embargo, por cuestiones de confidencialidad no se mostrará el código de la aplicación ni se hará una descripción detallada y técnica sobre su funcionamiento. En lugar de ello, se presentarán las etapas que atraviesa la aplicación durante su ejecución mediante un diagrama de flujo, realizando una somera descripción de cada una de ellas.

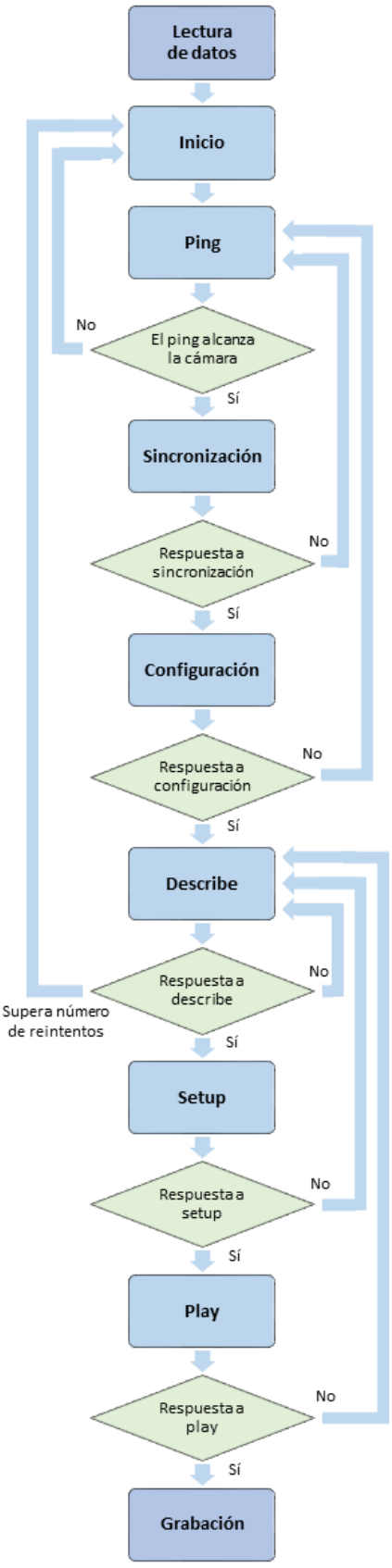

Figura 27. Diagrama de flujo de la aplicación.

En la figura anterior se presenta el diagrama de flujo de la aplicación. Como se puede observar, consta de nueve etapas. En los siguientes apartados se va a comentar cuál es la función principal de cada una de estas etapas.

5.3.1. Etapa "Lectura de datos"

En esta etapa se inicializan los datos relacionados con la cámara. Estos datos deben leerse del fichero de configuración en el cual se han especificado. En concreto, los datos a leer son:

- Número del tranvía.
- Dirección IP de la cámara.
- Fotogramas por segundo de la cámara.
- Resolución de la cámara, especificando altura y anchura.
- Modelo de la cámara, que en este caso es Everfocus.
- Códec de vídeo.
- URL de grabación a utilizar en las peticiones RTSP.
- Nombre de la cámara, necesario para generar el nombre de los ficheros de vídeo.
- Nombre del coche del tranvía, utilizado en la generación del nombre de los ficheros de vídeo.

5.3.2. Etapa "Inicio"

Es la etapa encargada de preparar las principales variables necesarias en la aplicación. Se establece el número de secuencia para la comunicación RTSP, se abren los *sockets* necesarios, se inicializan los datos de la transmisión con los que se rellenará el objeto moov del fichero MP4 y se inicializan los relojes, los temporizadores y los *flags*.

5.3.3. Etapa "Ping"

En esta etapa se envía un *ping* a la cámara para comprobar si está operativa y si se puede alcanzar. Se separa en una etapa independiente debido a que en etapas posteriores se retorna aquí si se producen pérdidas.

Si con el *ping* no se alcanza la cámara, la aplicación vuelve al estado de "Inicio". En caso contrario, se avanza a la siguiente etapa.

5.3.4. Etapa "Sincronización"

La función principal de esta etapa es sincronizar la hora de la cámara IP. Sin embargo, también se incluye en esta etapa la activación del OSD para mostrar la fecha y la hora superpuestas al vídeo.

Se puede elegir hacer la sincronización de la hora de manera manual o por NTP, tal y como se muestra en la figura 28. En ambos casos, primero debe configurarse el DST, activándolo o no en función de la fecha actual.

En el ajuste manual se envía la configuración de la hora directamente. Mientras que al utilizar NTP, primero se envía la petición indicando que se va a utilizar y posteriormente se define el servidor NTP.

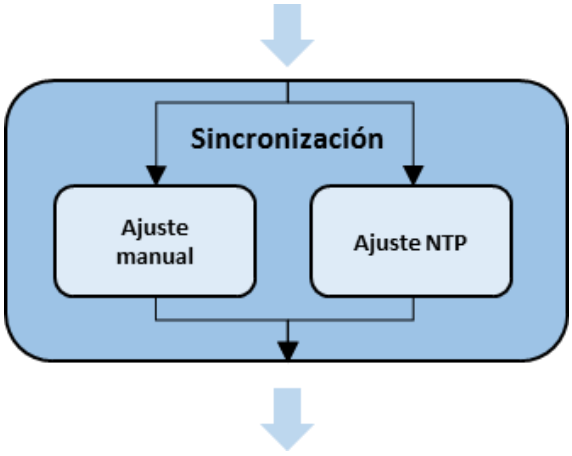

Figura 28. Etapa de sincronización.

Al final de esta etapa se debe comprobar la respuesta de la cámara. Si al analizar la respuesta en formato XML se detecta el campo "*statusCode*" y el campo "*statusString*" a "1" y "OK" respectivamente, significa que ha tenido éxito y se puede avanzar a la siguiente etapa. En caso contrario se vuelve a la etapa "Ping".

5.3.5. Etapa "Configuración"

En esta etapa se configura el canal de vídeo con las características que se deseen, como se ha visto en apartados anteriores de este capítulo.

Si al analizar la respuesta de la cámara se detecta que se han modificado los valores correctamente (campo "*statusCode*" a "1" y "*statusString*" con valor "OK"), se avanza a la siguiente etapa del programa. En caso contrario se retorna a la etapa "Ping".

5.3.6. Etapa "Describe"

Esta etapa se encarga de enviar un mensaje RTSP con el método DESCRIBE. De esta manera se obtiene toda la información útil para la conexión RTSP y para la inicialización de parámetros.

En caso de fallo en la comunicación o en la respuesta de la cámara, se vuelve a iniciar esta misma etapa. Si se supera el número de reintentos de esta etapa, se vuelve a la etapa "Inicio". En caso de éxito en la respuesta (campo "*statusCode*" a "1" y "*statusString*" con valor "OK"), se procesa la misma y se obtienen algunos valores para los objetos que conforman el fichero de vídeo MP4.

5.3.7. Etapa "Setup"

En esta etapa se envía un mensaje RTSP con el método SETUP, creando la sesión RTSP para la transmisión de vídeo desde la cámara.

Si la respuesta de la cámara al mensaje es correcta (campo "*statusCode*" a "1" y "*statusString*" con valor "OK"), se continua a la siguiente etapa. En caso contrario se vuelve al estado "Describe".

5.3.8. Etapa "Play"

La función de esta etapa es enviar un mensaje RTSP que contiene el método PLAY. De esta manera, la cámara IP sabe que puede comenzar la transmisión del vídeo en tiempo real.

Si la respuesta a este mensaje no es correcta, se retrocede al estado "Describe". En caso de recibir una respuesta correcta de la cámara (campo "*statusCode*" a "1" y "*statusString*" con valor "OK"), se avanza al último estado.

5.3.9. Etapa "Grabación"

En esta etapa se realiza la grabación del vídeo propiamente dicha. Esta grabación comienza con la transmisión de los fotogramas I y los fotogramas P desde la cámara IP y su recepción en el equipo de grabación.

En la figura 29 se muestra una captura de Wireshark resumida de los paquetes recibidos durante la transmisión de vídeo por parte de la cámara. En primer lugar transmite el fotograma I o *keyframe* dividido en varios paquetes. El inicio y fin de este fotograma se marca con los paquetes "*FU-A Start: IDR-Slice*" y "*Mark FU-A END*" respectivamente. FU-A hace referencia a un paquete que es un fragmento de una unidad superior o *fragmentation unit* en inglés.

Tras el fotograma I se transmiten una serie de fotogramas P, los cuales también se dividen en paquetes. De manera similar a los *keyframes*, el inicio y fin de estos fotogramas se marca con los paquetes "*FU-A Start: non-IDR-Slice*" y "*Mark FU-A END*" respectivamente.

```
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50677, Time=3676831712 FU-A Start:IDR-Slice
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50678, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50679, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50680, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50681, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50682, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50683, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50684, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50685, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50686, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50687, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50688, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50689, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50708, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50709, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50710, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50711, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50712, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50713, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50714, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50715, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50716, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50717, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50746, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50747, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50748, Time=3676831712 FU-A
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50749, Time=3676831712 FU-A
1114 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50750, Time=3676831712, Mark FU-A End
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50751, Time=3676835320 FU-A Start:non-IDR-Slice
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50752, Time=3676835320 FU-A
 783 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50753, Time=3676835320, Mark FU-A End
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50754, Time=3676838927 FU-A Start:non-IDR-Slice
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50755, Time=3676838927 FU-A
 744 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50756, Time=3676838927, Mark FU-A End
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50757, Time=3676842534 FU-A Start:non-IDR-Slice
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50758, Time=3676842534 FU-A
 943 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50759, Time=3676842534, Mark FU-A End
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50760, Time=3676846142 FU-A Start:non-IDR-Slice
1490 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50761, Time=3676846142 FU-A
 892 PT=DynamicRTP-Type-96, SSRC=0x5238, Seq=50762, Time=3676846142, Mark FU-A End
```

Figura 29. Captura de fotogramas con Wireshark.

Cada fotograma I y sus correspondientes fotogramas P se almacenan en un *buffer*, formando una secuencia de fotogramas. Cuando en el *buffer* ya se ha guardado el *keyframe* y sus fotogramas P, este se transfiere a una matriz. Sin embargo, en caso de no recibir el *keyframe*, se descartan todos sus fotogramas P.

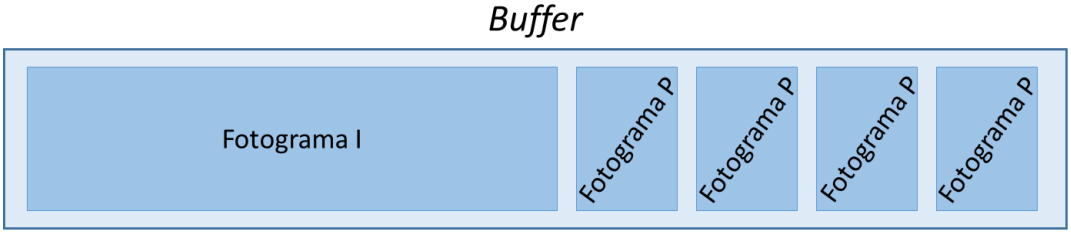

Figura 30. *Buffer* con una secuencia de fotogramas.

La secuencia de fotogramas guardada en el buffer se transfiere a una matriz cuando está completa, almacenándose cada secuencia en una fila independiente. Esta matriz en memoria es de tamaño configurable.

Transcurrido un tiempo definido o si la matriz se llena y contiene al menos un *keyframe*, la matriz se vuelva a disco formando el objeto mdat del fichero de vídeo MP4. Además, se crean los objetos ftyp y moov con los datos recogidos desde el inicio de la comunicación. De esta manera se genera un fichero de vídeo con formato MP4 y con el nombre adecuado.

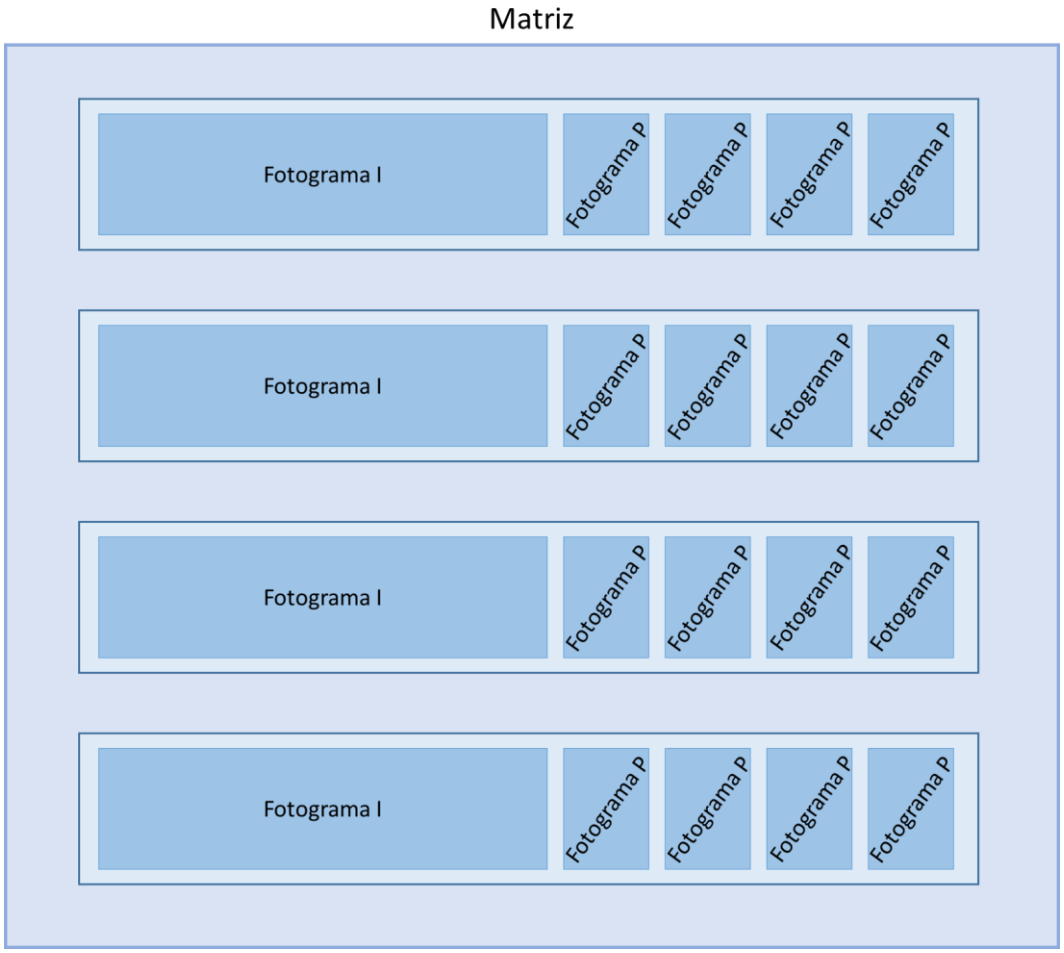

Figura 31. Matriz de fotogramas.

En caso de producirse pérdidas de paquetes durante la grabación de vídeo, si estas superan un determinado umbral, se cierra el fichero de vídeo y se reinicia la conexión.

Por último, cuando se manda a la aplicación cerrarse, primero guarda todos los ficheros de vídeo que pudiesen estar en proceso de creación y, después, es posible configurarla para que apague el equipo de grabación al cerrarse.

Capítulo 6

Manual de usuario

En este capítulo se presenta un breve manual de usuario, en el cual se explican las principales características que se pueden observar al ejecutar la aplicación de grabación en un equipo.

Durante el desarrollo de la integración del nuevo modelo de cámaras en la aplicación, esta se ha ejecutado desde el propio Eclipse para hacer pruebas, ya que permite depurar en caso de fallo. Sin embargo, también es posible generar un ejecutable que lance la aplicación.

Al ejecutar la aplicación de grabación, lo primero que se aprecia a simple vista es la apertura de un terminal clásico de Linux. En este terminal se muestran mensajes similares a los que se guardan en los ficheros de *log*. De esta manera, si la aplicación se bloquea en algún punto concreto, se puede ver el fallo a simple vista sin necesidad de consultar los ficheros de *log*. El aspecto de este terminal es el mostrado en la siguiente figura (se han ocultado líneas por cuestiones de confidencialidad):

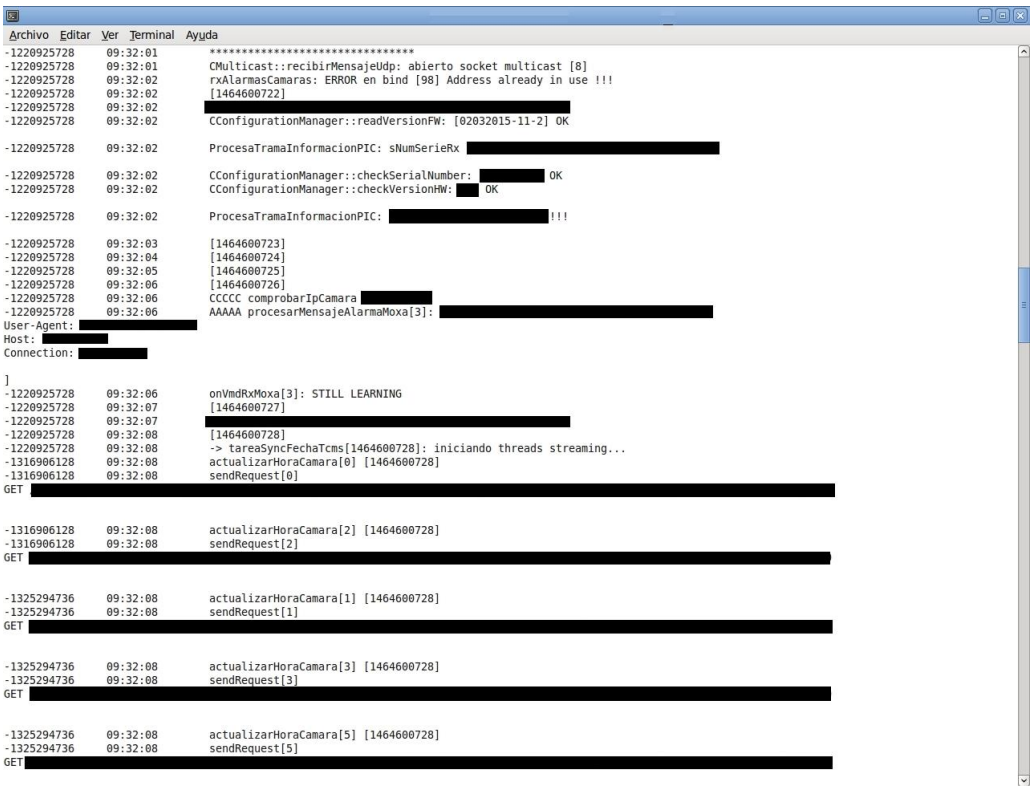

Figura 32. Terminal de ejecución de la aplicación.

Si se accede al disco duro que se emplea para almacenar los vídeos grabados de las cámaras, se puede observar que se genera un árbol de carpetas. En primer lugar se crea una carpeta llamada “capt”, la cual contiene los vídeos de los trenes. Dentro de esta carpeta se genera otra cuyo nombre es el identificador del tren. En la siguiente imagen el tren tiene un identificador igual a “2”:

Figura 33. Carpeta creada con nombre igual al identificador del tren.

La carpeta con nombre igual al identificador del tren contiene otras tres sub-carpetas que completan la estructura en árbol, tal y como se aprecia en la figura mostrada a continuación:

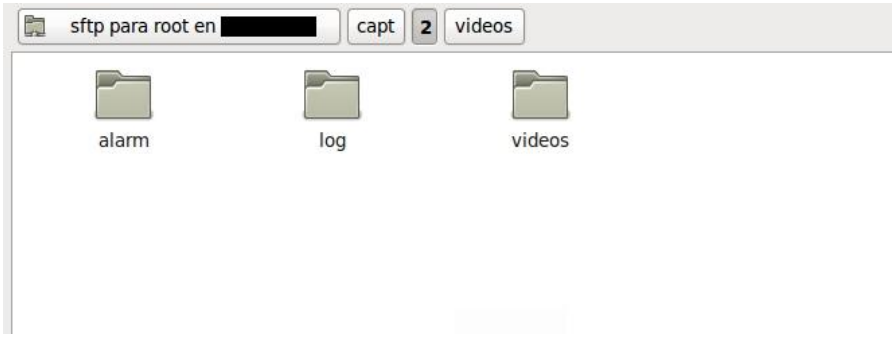

Figura 34. Carpeta con subcarpetas "alarm", "log" y "videos".

La primera carpeta se denomina “alarm” y contiene ficheros con las alarmas generadas cada día en las cámaras, como la alarma de *tamper* generada cuando se intenta tapar o manipular la cámara. En el sistema desarrollado en este Trabajo no se ha implementado ninguna alarma.

La siguiente carpeta es “log”, la cual contiene los ficheros de *log* generados cada día. De esta manera, en caso de que se reporta un error se puede consultar el

fichero del día correspondiente y buscar cuál ha sido el fallo. El nombre de los ficheros de *log* es muy similar al de los ficheros de vídeo: LOG _ [ID_tren] _ [Fecha_creación] _ [Hora_creación] _ [Nombre_equipo_grabador]. En la siguiente figura se muestra un ejemplo del contenido de esta carpeta:

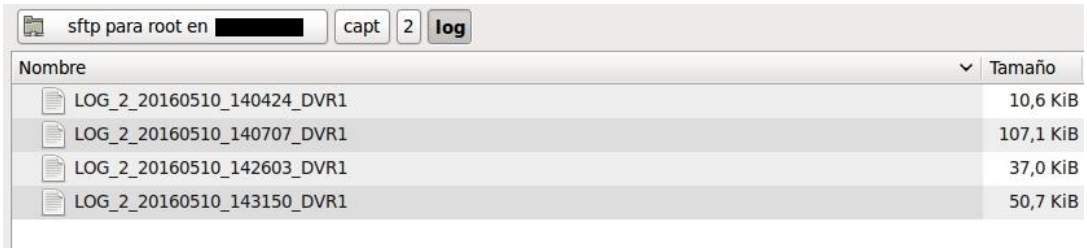

Figura 35. Carpeta "log".

En tercer lugar se encuentra la carpeta “videos”. Esta carpeta contiene a su vez una carpeta por cada día de grabación, de manera que todos los vídeos estén agrupados por fechas. Tal y como se aprecia en el ejemplo de la figura 36, el nombre de estas carpetas se corresponde con la fecha de grabación de los vídeos.

Figura 36. Carpeta "videos".

Finalmente, en estas últimas carpetas se encuentran los vídeos grabados de las cámaras IP. El formato de nombre de los ficheros de vídeo es el siguiente: [ID_tren] _ [Fecha_creación] _ [Hora_creación] _ [Vagón] _ [Número]. En la figura 37 se muestra un ejemplo del contenido de una de estas carpetas.

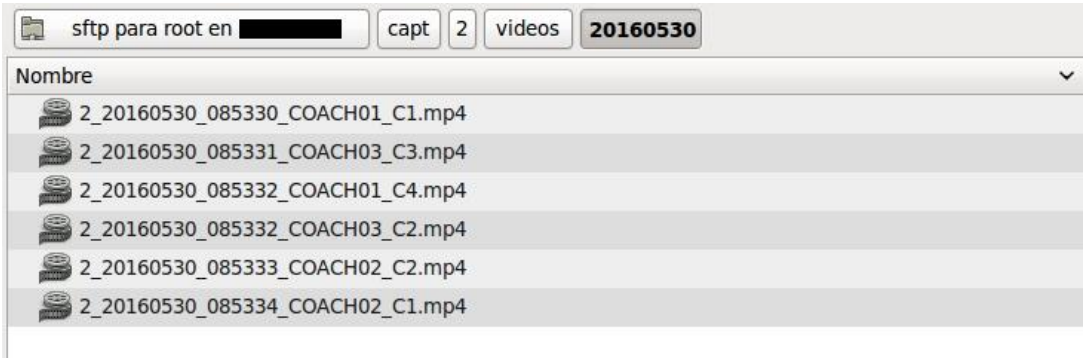

Figura 37. Carpeta con los vídeos grabados.

Se puede observar que el inicio de los ficheros de vídeo es casi simultáneo y que corresponden a cámaras con distinto número y ubicadas en distintos vagones.

Capítulo 7

Pruebas

En este capítulo se explicarán los principales problemas acaecidos a lo largo del desarrollo de la aplicación, así como los pasos seguidos para su solución. También se detallarán los problemas detectados durante las pruebas de grabación del sistema de vídeo vigilancia y sus soluciones.

7.1. Problemas detectados durante el desarrollo de la aplicación

En este apartado se detallarán los problemas más importantes surgidos en el desarrollo del sistema, asimismo se analizará la forma utilizada para solucionarlos.

- ***Problema:*** **Fallo en la configuración del servidor NTP**

- *Síntomas*: Se intentó establecer el servidor NTP para la configuración de hora automática, utilizando el método PUT sobre el recurso con URI "*/PSIA/System//NTPServers*", el cual se indicaba en el manual de programación CGI. Sin embargo, el servidor NTP no se modificaba.

- *Causas*: Utilizando el programa Wireshark se pudo comprobar que la cámara respondía, pero esa respuesta tenía el campo "*statusCode*" a "4" y el campo "*statusString*" contenía "*Invalid Operation*". Esto indicaba que no se podía operar sobre ese recurso.

- *Soluciones consideradas*:

  a) Obtener el recurso correspondiente a la URI utilizando el método GET, de esta manera se buscaba comprobar si la estructura del XML enviado en la petición de modificación era correcta. La respuesta a esta nueva petición era idéntica a cuando se utilizaba el método PUT, por lo que se descartó que el problema se debiese a un mal formato del contenido XML.

b) Realizar la configuración del servidor NTP a través de la interfaz web, para comprobar si de esta manera funcionaba correctamente. Se observó que con este procedimiento funcionaba bien, así que se descartó que el problema estuviese en la configuración NTP de la propia cámara.

- *Solución final*: Se contactó con el proveedor, el cual confirmó que el problema se debía a una errata en el manual de programación CGI. La URI que debía utilizarse era "*/PSIA/System/time/NTPServers*".

➢ ***Problema:*** **Fallo en el ajuste de la hora**

- *Síntomas*: Realizando el ajuste de hora manual cuando la fecha, la hora y la zona horaria eran incorrectas, se observó que sólo se modificaba la fecha y la zona horaria, permaneciendo la hora sin ajustar.

- *Causas*: No se podía modificar a la vez la hora y la zona horaria de la cámara, ya que si se hacía por separado funcionaba correctamente.

- *Soluciones consideradas*:

a) Utilizar el ajuste de hora automático por NTP para modificar los tres parámetros al mismo tiempo. Sin embargo, se comprobó que el problema persistía utilizando este método.

- *Solución final*: Se comprobó que enviando dos peticiones iguales seguidas, se modificaba correctamente la fecha, la hora y la zona horaria de la cámara. El proveedor confirmó que esta era la única manera posible de hacer el ajuste.

➢ ***Problema:*** **Problema en la interfaz web de la cámara**

- *Síntomas*: Al utilizar la interfaz web de la cámara, no se mostraban los botones de *zoom*, captura de imagen, grabación de vídeo en la memoria SD y reproducción de vídeo desde la SD.

- *Causas*: La interfaz web no mostraba todas las opciones en el navegador Chrome [32] y en Internet Explorer 11 [33] ni siquiera se mostraba la interfaz.

- *Soluciones consideradas*:

  a) Utilizar el navegador Firefox [34] para probar la interfaz web. Esta solución no es correcta ya que persiste el mismo problema que usando Chrome.

  b) Actualizar el *firmware* de la cámara como consejo del proveedor. Sin embargo, esto no solucionó el problema.

  c) Buscar versiones anteriores de los navegadores para comprobar si el problema eran las actualizaciones. Pero tras probar las versiones antiguas que se lograron descargar de algunos navegadores, se llegó a la conclusión de que no era la solución porque se desconocía la versión concreta con la que funcionaba la interfaz y de Internet Explorer no se encontraron versiones anteriores a la 10.

- *Solución final*: Se instaló la extensión IE Tab [35] en el navegador Chrome, que permitía emular diferentes versiones de Internet Explorer. En concreto, la interfaz web de la cámara mostraba todos los controles al utilizar Internet Explorer 8.

7.2. Problemas detectados durante las pruebas de grabación

Tras desarrollar la integración del nuevo modelo de cámaras en el sistema de grabación de vídeo, se realizaron pruebas consistentes en dejar el sistema grabado para detectar fallos y errores. A continuación se detallan los problemas detectados y las soluciones a los mismos.

- ***Problema:*** **Error en la etapa de configuración**

• *Síntomas*: La aplicación se detenía tras enviar la petición de configuración del canal de vídeo.

• *Causas*: Se generaba un error en la función encargada de generar los *logs* al guardar la petición que se acababa de enviar.

• *Soluciones consideradas*:

a) Aumentar el tamaño de las variables en las que se guardaban las peticiones para generar posteriormente el fichero de *log*, ya que la petición de configuración del canal de vídeo es bastante más extensa que el resto de peticiones, como se ha visto en el capítulo anterior. Sin embargo, esta medida no resolvió el problema.

• *Solución final*: Depurando el programa se observó que el tamaño del *buffer*, encargado de escribir las líneas correspondientes en el fichero de *log*, no era suficientemente grande. En este *buffer* se almacena la fecha y la hora, la función que llama al *log* y, en este caso, la petición de configuración completa. Al aumentar el tamaño de este *buffer* se solucionó el problema.

➢ ***Problema:*** **Fallo en el nombre de los ficheros de vídeo**

- *Síntomas*: La hora presente en el nombre de los ficheros de vídeo no coincide con la hora de inicio mostrada en los propios vídeos.

- *Causas*: Los tiempos controlados por la aplicación, a partir de las marcas de tiempo de los paquetes, se van descompensando respecto a los tiempos reales de la cámara.

- *Soluciones consideradas*:

  a) Disminuir el tiempo entre las actualizaciones de hora empleando NTP. Se observó que continuaban sin coincidir los tiempo, por lo que el problema no estaba en la sincronización de la hora en la cámara.

- *Solución final*: Se aumentó la precisión de las variables que operan con las marcas de tiempo de los paquetes, de manera que el resultado de las operaciones fuese más exacto al aumentar el número de dígitos que utilizaba.

➢ ***Problema:*** **Problema al grabar tras modificar la hora**

- *Síntomas*: En la situación en la que la cámara tiene la hora desconfigurada, al lanzar la aplicación de grabación de vídeos primero se genera un vídeo corto que muestra la hora incorrecta y, después, se generan vídeos del tamaño adecuado con la hora correcta.

- *Causas*: La grabación de vídeo comenzaba antes de que la cámara IP llegase a sincronizar la hora.

- *Soluciones consideradas*:

  a) Actualizar la hora de la cámara por medio del ajuste manual, eliminado así el posible retraso que pudiese haber al tener que sincronizarse la

cámara con el servidor NTP. Sin embargo, esta medida no solucionó el problema.

- *Solución final*: La cámara tarda aproximadamente dos segundos en cambiar la hora mostrada en el vídeo, tras haber recibido una petición de ajuste de hora. Es por esto que se introdujo en la aplicación un tiempo de espera de dos segundos entre el ajuste de la hora y el comienzo de la grabación de vídeos.

Capítulo 8

Presupuesto económico

En este capítulo se presenta el presupuesto económico de la integración realizada. Se especificarán todos los tipos de gastos asociados al desarrollo del sistema de grabación, tanto *software* como recursos *hardware*.

Se debe tener en cuenta que existen diversos factores determinantes para la estimación del presupuesto económico. El más importante de estos factores es el número de horas empleadas en el proyecto. Cabe destacar que en este trabajo concreto, aproximadamente la mitad de la duración del proyecto se empleó en comprender el funcionamiento completo del sistema de vídeo vigilancia utilizado hasta el momento. Sin embargo, estas horas no se incluirán en el presupuesto, puesto que es conocimiento adquirido y útil para hacer nuevas ofertas al cliente. Por lo tanto, se presupuestarán únicamente las horas empleadas en comprender todas las posibilidades del nuevo modelo de cámaras, desarrollar la integración y probar la nueva aplicación.

El presupuesto económico para el desarrollo de la integración, suponiendo que lo ha realizado un trabajador autónomo, es el detallado en la siguiente tabla:

Estimación del presupuesto económico		
Mano de obra	25 €/hora * 240 horas	6000 €
Cuota de autónomo	267.03 €/mes * 2 meses	534.06 €
***Software* libre**	0 €/mes	0 €
Coste del ordenador portátil	Coste total: 600 € Amortización fiscal: 12.5 €/mes	18.75 €
Coste del switch PoE	Coste total: 80 € Amortización fiscal: 1.67 €/mes	2.5 €
Gastos corrientes	Alquiler, luz, conexión a Internet, teléfono, calefacción y agua	420 €
Beneficio	5%	348.77 €
	TOTAL	**7324.08 €**

Tabla 4. Estimación del presupuesto económico.

Antes de comenzar con la explicación de cada coste del presupuesto, se deben aclarar tres aspectos importantes para el mismo. En primer lugar, se ha establecido un total de 240 horas para tener la solución desarrollada y probada, esto es un mes y medio a una media de 20 días hábiles al mes, realizando jornadas laborales de 8 horas al día. En segundo lugar, en las tablas de amortización fiscal de la Agencia Tributaria, se permite aplicar un 25% a los equipos de procesos de información, que en este desarrollo serían el ordenador portátil y el *switch* PoE. Por último, el coste de la cámara Everfocus no se incluye en el presupuesto por cuestiones de confidencialidad, así que se hace la suposición de que el cliente la suministra junto con el código del sistema de grabación utilizado hasta el momento.

A continuación se va a explicar cada partida del presupuesto:

- La mano de obra se ha calculado como el salario aproximado de un ingeniero junior (25€/hora) por las 240 horas empleadas en el estudio de la cámara, desarrollo y pruebas del sistema. Este salario constituye el mayor coste en el presupuesto económico.

- La cuota de autónomo es el coste mensual de la seguridad social dado de alta como autónomo. La cuota es de 267.03€ al mes y se debe pagar por meses, por lo que en este caso se deben pagar dos meses.

- Todo el *software* utilizado es gratuito, así que no se deben tener en cuenta gastos de licencias de los programas utilizados.

- Se ha estimado un coste del ordenador portátil de 600€, al cual se puede aplicar el 25% de amortización anual. En el presupuesto se incluye la parte proporcional al mes y medio que dura el desarrollo completo.

- El coste del *switch* PoE se estima de manera idéntica al coste del ordenador portátil.

- Los gastos corrientes son una estimación del coste que supondría el alquiler del local donde se situaría la oficina, la luz, la conexión a Internet, el teléfono, la calefacción y el agua durante los 42 días que corresponderían al mes y medio de desarrollo.

- El beneficio se ha estimado como el 5% de todos los demás costes y sería la retribución que se obtendría por el desarrollo de esta integración.

Finalmente, la estimación del presupuesto económico para el desarrollo de la nueva aplicación de vídeo vigilancia IP elaborada en este Trabajo Fin de Máster, es de 7324.08€.

Capítulo 9

Conclusiones y líneas futuras

En este capítulo se va a realizar un breve resumen del trabajo desarrollado, y en el cual se asientan las conclusiones extraídas de su realización, así como las líneas futuras de trabajo que puedan tomar como base la labor realizada en este Trabajo Fin de Máster.

9.1. Conclusiones

En los últimos años cada vez resulta más importante garantizar la seguridad de las personas en nuestra sociedad. Esta necesidad de seguridad a nivel mundial se puede observar en diversos ámbitos. En el caso de los grandes medios de transporte públicos esta necesidad resulta aún más prioritaria, pudiendo resultar útil el empleo de sistemas de vídeo vigilancia.

En este Trabajo Fin de Máster se ha integrado un nuevo modelo de cámaras en un sistema de vídeo vigilancia IP. La nueva aplicación trata de ampliar las compatibilidades con nuevos modelos de cámaras IP del sistema de grabación ofertado hasta ahora por GMV.

La solución que se plantea en este trabajo se apoya en el empleo de cámaras IP y tecnología digital. En concreto se basa en los protocolos RTP y RTSP, la codificación H.264 y el formato de vídeo MP4. De esta manera se ha obtenido una única aplicación capaz de grabar el vídeo de diferentes modelos de cámaras IP. Además, el sistema vídeo vigilancia se puede integrar con múltiples sistemas que también emplean las redes IP.

Con la integración del nuevo modelo de cámaras Everfocus en la solución utilizada hasta el momento, se logra un ahorro importante en costes. En concreto, por cada cámara IP se ahorra 106€, que es un 44% de ahorro respecto al modelo anterior. Trasladando esto a un ejemplo de un tranvía con cinco cámaras, supone un ahorro de 530€.

La aplicación de grabación de vídeos presentada en este trabajo permite ofrecer un sistema de vídeo vigilancia IP totalmente funcional, integrable con nuevos sistemas y funcionalidades y accesible a un mayor número de clientes gracias a la disminución de su precio.

9.2. Líneas futuras

Existen dos líneas futuras principales que se pueden definir como continuación de este trabajo. La primera consiste en mejorar las funcionalidades del sistema añadiendo nuevas posibilidades. La segunda, se trata de mejorar el sistema para extenderlo a otros sectores de la sociedad.

En cuanto a la primera línea enfocada a mejorar las funcionalidades del sistema, se puede:

- Configurar la detección de movimiento, de manera que la cámara IP genere alertas y estas se envíen a los sistemas adecuados.

- Incluir la grabación del audio captado por las cámaras Everfocus junto con el vídeo.

- Implementar el reinicio de la cámara desde la aplicación de grabación para cuando se producen fallos graves.

- Solicitar la información de la cámara con el método GET desde la aplicación y generar un fichero con la información de todas las cámaras IP del sistema.

- Implementar la comprobación y actualización del *firmware* de la cámara desde la aplicación.

En cuanto a la línea futura dirigida a mejorar el sistema de vídeo vigilancia para extenderlo a otros sectores, se puede:

- Integrarlo con modelos de cámaras IP que posean mejores prestaciones, de manera que sean cámaras adecuadas para el uso en grandes espacios públicos.

- Planificar la redundancia y seguridad del sistema, garantizando su funcionamiento en entornos críticos.

Bibliografía

Bibliografía

[1] GMV [En línea]. Recuperado de: http://www.gmv.com/es. [Último acceso: Mayo 2016].

[2] Everfocus [En línea]. Recuperado de: http://www.everfocus.com/. [Último acceso: Mayo 2016].

[3] Renfe [En línea]. Recuperado de: http://www.renfe.com/. [Último acceso: Mayo 2016].

[4] Talgo [En línea]. Recuperado de: http://www.talgo.com/index.php/es/home.php. [Último acceso: Mayo 2016].

[5] Renault [En línea]. Recuperado de: http://www.renault.es/. [Último acceso: Mayo 2016].

[6] Auvasa [En línea]. Recuperado de: http://www.auvasa.es/. [Último acceso: Mayo 2016].

[7] Moxa [En línea]. Recuperado de: http://www.moxa.com/. [Último acceso: Mayo 2016].

[8] The Eclipse Foundation, «Eclipse» [En línea]. Recuperado de: https://eclipse.org/. [Último acceso: Mayo 2016].

[9] Canonical Ltd., «Ubuntu» [En línea]. Recuperado de: http://www.ubuntu.com/. [Último acceso: Mayo 2016].

[10] Software in the Public Interest, Inc., «Debian» [En línea]. Recuperado de: http://www.debian.org/index.es.html. [Último acceso: Mayo 2016].

[11] Thin Multimedia, Inc., «MP4 Reader» [En línea]. Recuperado de: http://www.thinmultimedia.co.kr/products/MP4Reader.html. [Último acceso: Mayo 2016].

[12] Wireshark Foundation, «Wireshark» [En línea]. Recuperado de: http://www.wireshark.org/. [Último acceso: Mayo 2016].

[13] M. Hörz, «HxD» [En línea]. Recuperado de: https://mh-nexus.de/en/hxd/. [Último acceso: Mayo 2016].

[14] EverFocus Co., Ltd., «Hoja de especificaciones técnicas, EMN2120 nevio HD Series» 2013.

[15] V. Carli, «Valoración del CCTV como una Herramienta efectiva de manejo y seguridad para la resolución, prevención y reducción de crímenes» Centro Internacional para la Prevención de la Criminalidad, Montreal, 2008.

[16] Aventura Technologies, Inc, «Aventura Security Solutions by Design» 2009. [En línea]. Recuperado de: http://www.aventuracctv.com/newsletter/DOCS/Aventura_Newsletter_02_Analog_vs_IP_Cameras.pdf. [Último acceso: Mayo 2016].

[17] Axis Communications, «Convertir un sistema de CCTV analógico en uno de Vigilancia IP» 2003.

[18] A. Durresi y R. Jaun, «RTP, RTCP, and RTSP - Internet Protocol for Real-Time Multimedia Communication» 2005.

[19] H. Schulzrinne, S. Casner, R. Frederick y V. Jacobson, «RFC 3550. RTP: A Transport Protocol for Real-Time Applications» 2003.

[20] H. Schulzrinne, A. Rao, R. Lanphier, M. Westerlund y M. Stiemerling, «RFC 2326. Real Time Streaming Protocol 2.0 (RTSP)» 2014.

[21] CADYTEL, «¿Qué es la compresión H.264?» 2008.

[22] ITU-T, «Recommendation ITU-T H.264. Advanced video coding for generic audiovisual services» 2014.

[23] Y. Wang, R. Even, T. Kristensen y R. Jesup, «RFC 6184. RTP Payload Format for H.264 Video» 2011.

[24] Axis Communications, «Estándar de compresión de vídeo H.264. Nuevas ventajas para la videovigilancia» 2008.

[25] C. Concolato, «MPEG File Formats» 2005.

[26] ISO/IEC, «Information technology - Coding of audio-visual objects - Part 12: ISO base media file format» 2005.

[27] ITU-T, «Recommendation ITU-T J.124. Multiplexing format for multimedia webcasting» 2004.

[28] «MP4 Atom Parsing» [En línea]. Recuperado de: http://stackoverflow.com/questions/18436551/mp4-atom-parsing-where-to-configure-time. [Último acceso: Mayo 2016].

[29] GMV, «Documentación interna de GMV» 2014.

[30] A. Kapoulkine, «pugixml» [En línea]. Recuperado de: http://pugixml.org/. [Último acceso: Mayo 2016].

[31] EverFocus Co., Ltd., «CGI Programming. User Guide v1.0.4» 2014.

[32] Google, «Chrome» [En línea]. Recuperado de: https://www.google.com/chrome/browser/desktop/index.html. [Último acceso: Mayo 2016].

[33] Microsoft, «Internet Explorer» [En línea]. Recuperado de: http://windows.microsoft.com/es-es/internet-explorer/download-ie. [Último acceso: Mayo 2016].

[34] Mozilla, «Firefox» [En línea]. Recuperado de: https://www.mozilla.org/es-ES/firefox/products/. [Último acceso: Mayo 2016].

[35] Blackfish Software, LLC., «IE Tab» [En línea]. Recuperado de: https://www.ietab.net/. [Último acceso: Mayo 2016].