JITEL 2017
XIII Jornadas de Ingeniería Telemática
27-29 Septiembre 2017, Valencia (España)

Actas de las XIII Jornadas de Ingeniería Telemática (JITEL 2017), Valencia (España), 27-29 de Septiembre de 2017

ISBN: 978-84-9048-595-8
DOI: http://dx.doi.org/10.4995/JITEL2017.2017.6591

Mitigando Efectos Adversos de Interrupciones del Servicio de Video-vigilancia del Hogar en Clientes WiFi inalámbricos

Tatiana Gualotuña[1], Elsa Macías[2], Alvaro Suárez[2], Efraín R. Fonseca C.[1], Andrés Rivadeneira[1]

[1]Departamento de Ciencias de la Computación,
[2]Departamento de Ingeniería Telemática (DIT),
[2]Instituto de Ciencias y Tecnologías Cibernéticas (IUCTC)

[1]Universidad de las Fuerzas Armadas (ESPE)

[2]Universidad de Las Palmas de Gran Canaria (ULPGC)

[1]Av. General Rumiñahui S/N y Paseo Escénico Santa Clara. Sangolquí - Ecuador.

[2]Edificio de Electrónica y Telecomunicación – Campus universitario de Tafira – 35017 Las Palmas de G.C.

tmgualotunia@espe.edu.ec, erfonseca@espe.edu.ec, alvaro.suarez@ulpgc.es, elsa.macias@ulpgc.es, avrivadeneira@espe.edu.ec

***Resumen*- Actualmente los sistemas de videovigilancia en el Hogar se pueden combinar con sensores para formar un sistema de muy bajo coste y fácil manejo por parte del usuario final. Un componente importante en este sistema es el servidor de video streaming en tiempo real a clientes Web que usan *Wireless Fidelity*. El servidor se puede instalar en *open hardware* como el *Raspberry Pi* y se puede ayudar de sensores *arduino* para detectar alarmas de intrusión o condiciones domóticas adversas en el Hogar. Sin embargo, el *Wireless Fidelity* tiene conocidos problemas que provoca interrupciones esporádicas e impredecibles del servicio de video y de poca duración que disparan la pérdida de *frames* de video clave para observar el estado del Hogar en un momento dado, mientras el cliente se mueve (después de recibir una alarma). Mitigar los efectos adversos de estas interrupciones es una tarea complicada que hemos trabajado durante años. La novedad es que ahora construimos un sistema de *open hardware embebida* y de bajo coste (con utilidad práctica a los ciudadanos), y software libre íntegramente basado en servicios Web que es interoperable y basado en ontologías (para incluir decisiones sofisticadas, *smart*, de reconectar el servicio interrumpido). El elevado valor de parámetros de calidad de experiencia de usuario avalan los buenos resultados alcanzados.**

***Palabras Clave*- Video-vigilancia, WiFi, Streaming, patrón de diseño software, RapBerry Pi, arduino.**

I. Introducción

La cantidad de robos en domicilios es uno de los parámetros que se usan para observar la Calidad de Vida de los ciudadanos de un país. En España [1], en 2016, el número de robos con fuerza está en los 4.800 y con violencia en 1.800; cifras parecidas a las del año 2015. Así mismo en Ecuador al 12% de la población le preocupa la delincuencia e inseguridad sobre otros problemas [2]. Estos temas han hecho que los servicios de telecomunicación basados en videovigilancia en el Hogar hayan prosperado enormemente en los últimos años al amparo de la jurisprudencia de protección de datos personales [3]. Esto es, la videovigilancia es una Industria madura que ha producido tecnologías y equipamiento propietario de muy diverso tipo. Los operadores de Telecomunicación han aprovechado este auge para proponer servicios al Hogar sobre las antiguas redes de acceso a Internet de cobre (*Digital Subscriber Line, DSL*) o bien las nuevas redes de alta velocidad basadas en Fibra óptica (*Fiber to the Home*). De esta forma, cualquier usuario puede recibir una alerta en el caso que se detecte una entrada inesperada a su Hogar e incluso podría inmediatamente recibir video de lo que está pasando dentro de su Hogar en tiempo real.

EDITORIAL UNIVERSITAT POLITÈCNICA DE VALÈNCIA

La popularidad de los sistemas de videovigilancia en el Hogar ha provocado la proliferación de sistemas de bajo coste de hardware y software libre que cualquier usuario puede desplegar (a base de tutoriales de la Web) en su propio Hogar y manejarlos a través de su conexión a Internet. El despliegue vertiginoso que ha tenido la *Internet of Things* (*IoT*) [4] ha convergido con los sistemas de videovigilancia propiciando nuevos servicios en los que además de la detección de intrusos inesperados se puede tener un control más completo (*domótico*) del Hogar observando como varían sus niveles de temperatura, humedad… El abaratamiento de los sistemas de computación y comunicación, en los últimos años ha sido espectacular. Computadores embebidos en una placa, como el *Raspberry Pi* [5] y plataformas de hardware abierto para diseño de redes de sensores como arduino [6] permiten un diseño muy barato de un sistema de videovigilancia, apoyado en sensores, de forma rápida y con un manejo muy sencillo por parte del usuario final.

Un componente importante de estos sistemas de videovigilancia es el servidor de video streaming en tiempo real. Estos servidores pueden ser desde *youtube* hasta servidores embebidos en las cámaras inalámbricas *Internet Protocol* (*IP*), o bien en los Raspberry Pi [7]. En todos estos casos el cliente puede recibir video directamente en un navegador Web.

Los sistemas de videovigilancia modernos son muy sofisticados y proponen un tratamiento del video enfocado, entre otros, a: a) el procesamiento de imágenes para obtener posiciones relativas de objetos soportando adversidades medioambientales y combinándolo con IoT para mejorar la adquisición de la información [8]. Nosotros no estamos interesados en la segmentación de objetos ni la afectación de condiciones medioambientales, sino en una recepción en tiempo real de imágenes que faciliten una primera visión por parte del usuario final que observa la intrusión a su Hogar. b) La implantación de sofisticados sistemas de compresión y codificación de objetos en un sistema de múltiples cámaras de videovigilancia para enviar únicamente información relevante al usuario final [9]. Nosotros tenemos un reducido número de cámaras en el Hogar con lo cual no tenemos necesidad de implantar un sistema de este tipo. c) Uso de computadores embebidos para alojar el sistema de video streaming conectado a una cámara de video a través de *Universal Serial Bus* (*USB*) y enviarlo a través de redes móviles. En [10] se utiliza un sistema empotrado basado en el procesador *ARM9* (de libre distribución), una tarjeta de red móvil 3G, una cámara USB que captura video utilizando el estándar H.264 y lo envía a un servidor de video. El usuario accede a él mediante un teléfono móvil Android. La construcción de un sistema empotrado como servidor y procesador de video es una opción importante debido a que se puede encontrar hardware y software de libre distribución manteniendo un esquema de comunicación de bajo costo que podría ser la solución para el Hogar. Sin embargo, en [10] no se enfocan, como nosotros en el video streaming en tiempo real. Nosotros usamos un computador embebido de propósito general (Raspberry Pi) por los buenos resultados de rendimiento que ofrece y su bajo coste.

El escenario que hemos contemplado es aquel en el que existe un Hogar controlado por sensores y una o varias cámaras de videovigilancia. Cuando los sensores detectan condiciones físicas fuera de un intervalo de normalidad entonces disparan la cámara de video que empieza a retransmitir en tiempo real a usuarios que disponen de teléfonos móviles *Wireless Fidelity* (*WiFi*). Estos clientes pueden estar en movimiento puesto que deben dirigirse hacia el Hogar. Durante ese movimiento se pueden dar situaciones en las que se interrumpa la recepción del video debido a múltiples factores revisados en [11]: comportamiento caótico de los canales radio, problemas de acceso al canal en WiFi, problemas de control incorrecto de congestión y flujo en protocolos de nivel de red y transporte en Internet para redes inalámbricas y mala gestión de la conexión de los servidores de video streaming que no detectan cuando un cliente ha perdido la conexión enviando frames de video que nunca serían recibidos por su destinatario. En este caso, un modelo simple de rendimiento demuestra que la *Quality of Service* (*QoS*) de la red se degrada enormemente, básicamente, porque por cada vez que se interrumpe el servicio, durante cierto tiempo, es necesario volver a iniciar la sesión de video streaming; pero además degrada enormemente la *Quality of Experience* (*QoE*) [12]. Nosotros en este artículo presentamos una solución novedosa a este problema que mitiga los efectos adversos de estas interrupciones de servicio en la QoE, a la vez que permite un mayor control de la seguridad en el Hogar.

Las novedades importantes de este trabajo son:

- Formulamos una solución interoperable, multiplataforma y basada íntegramente en la Web y agentes *Java Agent Development Framework* (*JADE*) [13] mediante el *addon* de JADE *Web Services Integration Gateway* (*WSIG*) [14] para transformar los comportamientos de los agentes JADE en servicios Web.
- Diseñamos ontologías para implantar comportamientos inteligentes de los agentes que controlan los sensores y las interrumpciones de servicio.
- Integramos el servidor de video en un Raspeberry Pi B+ e iniciamos la emisión de video si se producen alarmas en varios sensores que se han implantado en una plataforma arduino.
- Las medidas de la QoE demuestran el buen comportamiento y satisfacción de los usuarios entrevistados.

La estructura del artículo es la siguiente. En la Sección II revisamos brevemente un modelo sencillo de interrupciones de servicio y su efecto adverso en la QoE. En la Sección III se analiza el diseño del sistema de videovigilancia con control de las interrupciones de video streaming. En la Sección IV se presentan los resultados experimentales y los valores obtenidos para la métrica de *Mean Opinion Score* (*MOS*) para medir la QoE y finalmente presentamos algunas conclusiones y trabajo futuro.

II. LAS INTERRUPCIONES DEL SERVICIO DE VIDEO STREAMING

En la Fig. 1 se muestra un esquema de los componentes (modelo de cajas negras) principales de nuestro sistema de videovigilancia: los sensores, procesador de alarma de los sensores, el servidor de videostreaming y el cliente. El objetivo es introducir un modelo matemático sencillo que de idea de la problemática de las interrupciones del servicio y los efectos adversos que tiene en la QoE.

A grandes rasgos, el sistema de videovigilancia tiene las siguientes características:

- Los sensores están continuamente enviando datos sensados (con un periodo de muestreo determinado) al procesador de alarmas.
- La cámara de video se pone en funcionamiento una vez el procesador fusione los datos de los sensores y determine que se ha producido una alarma. En ese momento comienza la sesión de videostreaming con el cliente. A la vez, se para el procesamiento de nuevos datos sensados, hasta que se resuelva la alarma actual mediante una acción concreta del usuario (cliente).
- El servidor de video streaming recibe frames de la cámara de video y los almacena en un archivo (buffer) para posterior análisis de las grabaciones.
- Al mismo tiempo que se van almacenando los frames de video en el archivo, envía una notificación al cliente indicando la existencia de una intrusión u otra condición domótica excepcional en el Hogar.
- En ese momento el cliente inicia una sesión de videostreaming con el servidor, recibiendo, en tiempo real, los frames de video que va produciendo la cámara.
- En un momento determinado el cliente para la emisión de video streaming de la cámara a través de su aplicación móvil. En este momento el procesador de alarmas volverá a fusionar datos sensados en busca de nuevas alarmas.

Centrando ahora la atención en el envío de video streaming podemos tener dos escenarios distintos: a) desde que inició la sesión de video streaming hasta que la cancela el usuario, no se produce ninguna interrupción del servicio. b) Se producen una o varias interrupciones de video de una duración suficiente cómo para que el servidor cierre la sesión de video streaming, debiéndose iniciar de nuevo la sesión (esto implica volver a iniciar una nueva sesión y comenzar a enviar el video almacenado desde el principio). Cada vez que se produzca el cierre una sesión se debe volver a empezar la retransmisión desde el principio.

Analicemos el impacto en la QoS y la QoE con un modelo sencillo abstrayendo detalles de la comunicación y observando únicamente la cantidad de tiempo y energía necesaria para enviar todo el video almacenado y de manera informal cuál sería la satisfacción del usuario. En la Fig. 2 se muestra un esquema que facilita la comprensión del modelo. En el caso ideal (que no haya ninguna interrupción), el cliente (*C*) tarda *Ts* unidades de tiempo (*u.t.*) en iniciar la sesión de videostreaming e indicar el comienzo de la recepción de paquetes de video (*SETUP*, *OK* y *PLAY*). Después de t_p u.t. recibe el primer paquete de video. A partir de ahí recibe un nuevo paquete después de cada d_i u.t. (retraso de llegada del paquete *i* respecto al paquete *i-1*). Por tanto, el tiempo óptimo (T_{opt}) que tarda *C* en recibir todo el video es el que se muestra en la Ec. 1:

$$T_{opt} = T_s + t_p + \sum_{i=2}^{n} d_i \qquad (1)$$

Donde *n* es el número total de paquetes que se envían, T_s es el tiempo de inicialización de la sesión y t_p es el tiempo de llegada de un paquete.

En el caso que se produzcan *m* interrupciones del video entonces el tiempo (T_{int}) que tarda en recibirse el vídeo es considerablemente mayor y está en función del número de interrupciones que se producen y su duración (Ti_j), según indica la Ec. 2:

$$T_{\text{int}} = (T_s + t_p)m + \sum_{j=1}^{m} Ti_j + \sum_{i=2}^{K_j} d_i + T_{opt} \qquad (2)$$

Donde el factor K_i es la cantidad de paquetes que se recibe antes de producirse la interrupción *i* y suponemos que al menos se recibe uno antes de cualquier interrupción.

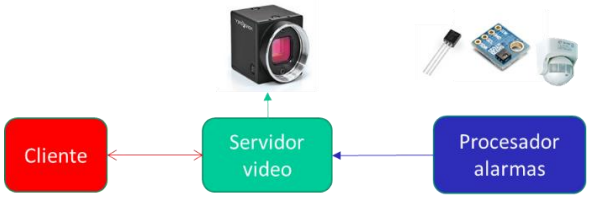

Fig. 1. Esquema del sistema de videovigilancia.

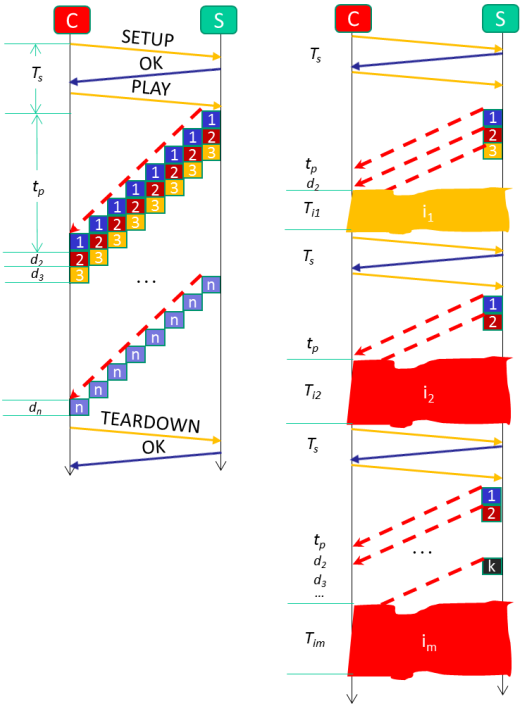

Fig. 2. Tiempos de comunicación sin y con interrupciones de servicio de video streaming.

Un efecto adverso importante es el número de veces que se retransmite un paquete de video que viene dado por la Ec. 3.

$$\sum_{i=2}^{K_i} d_i \tag{3}$$

En cada una de estas retransmisiones el paquete está ocupando el canal WiFi evitando que se pueda comunicar otros paquetes (aumentando por tanto el tiempo de contención). Además, si suponemos que el teléfono móvil del *C* tiene un gasto de *p* unidades de consumo de energía (u.e.), entonces el teléfono móvil estará haciendo un gasto (consumo extra ineficaz de batería) indicado en la Ec. 4.

$$\sum_{i=2}^{K_i} p \tag{4}$$

El efecto adverso más importante es la degradación de la QoE [15]. Es posible que el usuario acepte que se le retransmita una vez los mismos paquetes; pero no estaría dispuesto a que se lo retransmitieran más de tres veces. Y lo más probable es que decidiera no seguir reiniciando las sesiones de video streaming. Esto es importante porque en un sistema de videovigilancia como el que consideramos, precisamente un elemento distintivo es que el usuario pueda hacer una visualización del video lo más pronto posible mientras, por ejemplo, está camino a su Hogar.

Este no es un problema particular de nuestro sistema, sino que se produce en otros escenarios. Por ejemplo, en [16] se demuestra matemáticamente, que a nivel de enlace, las interrupciones cuya duración varía entre muchos segundos o pocos minutos producen *buffer underrun* y se propone mitigarlas mediante: control de *buffering*, monitorización de la red y la regulación de la inyección de paquetes de vídeo desde el Servidor. A nivel de enlace se puede tener un control de este tipo, pero cuando las interrupciones tienen cierta duración, el problema es que los servidores de videostreaming abortan la sesión de video y se debe recomenzar de nuevo como hemos indicado anterioremente. En [17] se analiza la optimización del uso de *caches* de video para lograr minimizar las interrupciones de video y más concretamente minimizar el T_s. En [18] se presenta un estudio de implantación de un sistema de videovigilancia sobre *Long Term Evolution* en el que se reconoce como una de las enseñanzas aprendidas que se debe proporcionar un sistema de *buffering* adecuado para poder soportar el videostreaming en tiempo real sin interrupciones de servicio, además de modificar el sistema de buffering de *Android*. Nosotros en [19] [20] analizamos el uso de varios buffers de video streaming y el empleo de proxies del servidor de video streaming que permitan monitotizar parámetros de la comunicación entre el cliente y el servidor para minimizar el T_s y las retransmisiones de paquetes, además de automatizar la reconexión de la sesión del videostreaming para evitar que el usuario lo deba hacer manualmente, reduciendo con ello el efecto adverso de las interrupciones de video streaming en la QoE. Nuestros esquemas pueden ser adaptados para construir la solución presentada en [18] teniendo en cuenta los mecanismos específicos de QoS.

III. Sistema de videovigilancia que mitiga las interrupciones de servicio de videostreaming

Dado que el sistema a implantar tiene una complejidad considerable, para implantar el software de la mitigación de los efectos adversos de las interrupciones de video streaming, en conjunción con el manejo de las alarmas de los sensores, se procedió a una especificación de los componentes del software basados en patrones de diseño de software [21]. En [19] se puede encontrar un análisis exhaustivo de la derivación formal de los mejores patrones para implantar cada componente. Hasta donde alcanza nuestro conocimiento esto representa una novedad importante en el diseño e implantación de sistemas de videovigilancia encaminados en la línea de sistematizar y estandarizar la parte de diseño software de servicios telemáticos, y va en la línea recomendada en [22] [23]. Como ejemplo, en la Fig. 3 se muestra los patrones utilizados para el diseño de los componentes de la Fig. 1. El patrón *Modelo Vista Controlador* (*MVC*) se ha usado para la interacción entre el cliente y el servidor

de video streaming al que se ha añadido un patrón *proxy* que interactúa con el servidor y el cliente. Para el manejo de los sensores se ha implantado el patrón *observador*. El *Modelo* maneja el buffer de video (almacenamiento temporal) que no puede ser transmitida en tiempo real porque se ha producido una interrupción. La *Vista* utiliza el patrón *Composite* que maneja la interfaz de usuario (*C*). Recibe un mensaje de alarma negocia la sesión de video y visualiza el video. Si se produce una interrupción, despliega un mensaje *reconectando* hasta que se reconecta automáticamente de nuevo que es cuando la visualización del video continúa. El *Controlador* utiliza los patrones *Proxy* y *Strategy*. El *proxy* solicita al *Observer* del servidor de procesado de alarmas (sistema externo) el estado de la información de los sensores procesada por él. Con ella genera alarmas y solicita que la cámara se ponga en funcionamiento. Cuando ocurre una interrupción de video invoca al patrón *Strategy* para que ejecute el algoritmo de tratamiento de la interrupción. Cuando se reestablece la sesión de video le solicita la ejecución del procedimiento de restablecimiento de la sesión. El *Observador* del sistema externo utiliza el patrón *Adapter* para transformar en servicios Web (JADE-WSIG) las acciones de los agentes inteligentes JADE. Estos agentes manejan ontologías para tratar de forma estándar los datos provenientes de los sensores. Estas ontologías facilitan el diseño e implantación de la inteligencia de los agentes para disparar alarmas. Aunque no se muestra en la Fig. 3, el video streaming *Proxy* también maneja ontologías de batería, posición y nivel de cobertura de los teléfonos móviles.

En [19] se presentan todos los diagramas de secuencia entre los objetos que implantan los componentes del sistema atendiendo a los patrones de diseño utilizados.

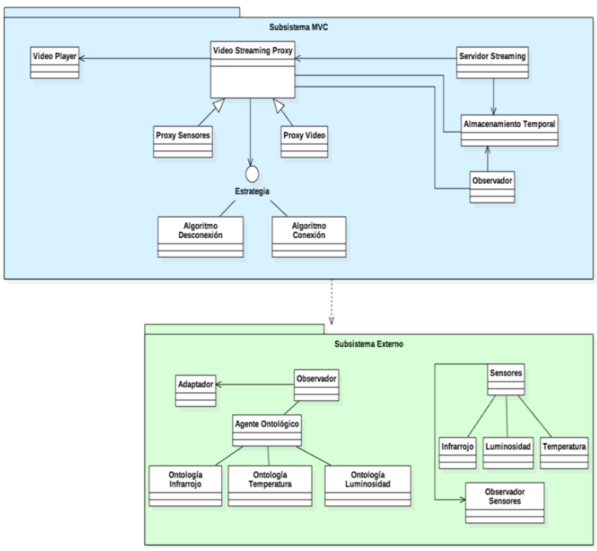

Fig. 3. Ejemplo de especificación de los patrones de diseño para el servidor de video, el cliente y el servidor de alarmas de los sensores.

A. Modelado de la solución a las interrupciones de video streaming

Con los mecanismos provistos en el sistema de videovigilancia, se mitigan los efectos adversos de las interrupciones de video streaming. En la Fig. 4 se muestra un esquema del tiempo que se invierte en el sistema de video streaming con el nuevo mecanismo. Se puede observar que ahora todos los paquetes de video se almacenan en el buffer la primera vez que se envían. Cuando se produce la i_1, los paquetes se siguen almacenando hasta completar todos los paquetes de video. Para evitar que el *C* envíe una orden de parar el video streaming se opta por almacenar el video durante una cantidad de tiempo prefijada pero configurable (entre 4 y 5 minutos, porque esta es la duración estimada que duran los atracos en el Hogar). En ese tiempo se almacenarían los *n* paquetes del video recogido por la cámara una vez el servidor de alarmas produce una nueva alarma.

Con el nuevo sistema el tiempo que se tarda en enviar el video completo está dado por la Ec. 5.

$$T_{buf} = T_s + t_p m + \sum_{j=1}^{m} Ti_j + \sum_{i=2}^{n-m} d_i \qquad (5)$$

Con lo cual, a partir de las Ec. 1, 2 y 5 se puede obsevar la relación de la Ec. 6.

$$T_{opt} < T_{buf} << T_{\text{int}} \qquad (6)$$

Lo que indica que el usuario ahorraría una cantidad de tiempo considerable en ver el vídeo en presencia de interrupciones en relación a si no se aplica nuestro método. Pero además, se le proporciona mayor comodidad al hacer que la reconexión se pueda manejar de forma automática. Esto indudablemente mejora la QoE.

Otro aspecto importante es el consumo de batería. Con este mecanismo se consigue minimizar el consumo extra de energía en retransmisiones. De forma indirecta esto también mejora la QoE debido a que el usuario dispone de mayor tiempo para estar conectado al sistema de videovigilancia. Por otro lado, es importante maximizar el tiempo de vida de la batería para asegurar una mayor tranquilidad del usuario de tal manera que no perciba que se puede desconectar del sistema y no saber que está pasado en en su Hogar.

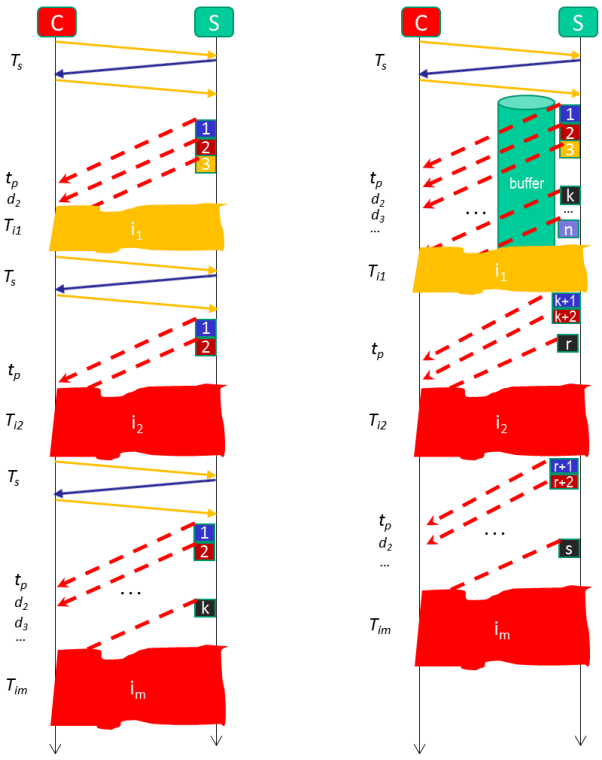

Fig. 4. Tiempos de comunicación con interrupciones de servicio de video streaming y con el nuevo método de mitigación de los efectos adversos de las interrupciones de servicio.

IV. Construcción del prototipo, implantación del software y resultados experimentales

A partir de la especificación basada en patrones de diseño software y ontologías se construyó un prototipo hardware y toda la programación del sistema completo: servidor de videostreaming, clientes inalámbricos y procesador de los datos sensados.

A. *Prototipo hardware*

Se construyó una maqueta física que emula un Hogar típico con varias habitaciones, garaje, escaleras… (Fig. 5). Dentro de esa maqueta se instalaron los sensores y la cámara de video para detectar alarmas de intrusión no deseadas y la grabación de las acciones correspondientes. En ella se instalaron 2 sensores de luz en la parte delantera y trasera del Hogar, 2 sensores de temperatura en la sala y cocina, 4 sensores infrarrojos para detección de movimiento (entrada delantera, trasera, cocina, baño) y 2 actuadores para enfriamiento (ventiladores). Las características técnicas de los sensores utilizados son:

- *Luz*: dimensiones 65x11x13 mm, serie Fotoresistor-BH1750, mediciones 1-65535 LX, muestreo 2s.
- *Temperatura digital* (*Ds28b20*): cada dispositivo tiene un código de serie 64-bit único almacenado en su *Read Only Memory* (*ROM*), capacidad multipunto que simplifica el diseño de las aplicaciones de detección de temperatura, puede ser alimentado desde la línea de datos. El rango de suministro de energía es de 3.0 V a 5.5 V. Mide temperaturas de -55 °C a + 125 °C (-67 °F a + 257 °F) ± 0,5 °C con precisión de -10 °C a + 85 °C. La resolución del termómetro es seleccionable por el usuario de 9 a 12 b y convierte la temperatura en códigos de 12 b en 750 ms (máximo).
- *Barrera Infrarroja*: fototransistor de 10.2 x 5.8 x 7 mm, distancia de operación pico: 2.5 mm, intervalo de operación para una variación de corriente de colector mayor al 20 %: 0.2mm a 15 mm, corriente de salida típica bajo prueba: Ic = 1 mA, filtro de bloqueo de luz ambiental, longitud de onda del emisor: 950 nm.

El procesado de los datos sensados se hizo mediante un microcontrolador arduino Atmega 328. Se utilizó una cámara de 20.7 Mpixels, flash LED, con sensor de ½.3”, geotagging, y estabilizador de imagen. Para implantar el servidor de videostreaming y otro software de control de almacenamiento… se usó una RaspBerry Pi B+ con un procesador de 900 MHz de cuatro núcleos de CPU ARM Cortex-A7, 500 GB de RAM, 4 puertos USB, 40 pines GPIO, interfaz de la cámara (CSI), interfaz de pantalla (DSI), ranura para tarjeta Micro SD y núcleo de gráficos VideoCore IV 3D.

B. *Implantación del software basado en componentes de software libre*

En la Fig. 6 se muestra un diagrama de los distintos componentes de software libre que se utilizaron para la implantación de los distintos componentes del sistema de videovigilancia.

Mediante *Raspduino* se hace interoperar a la plataforma arduino de sensores con la RaspBerry Pi B+ logrando que se pueda recoger datos de: el identificador del sensor, la medida que sensa y un sello de tiempo para ese valor sensado. Estos valores se envían a un *proxy* capaz de interactuar con agentes de JADE. En estos agentes se define el comportamiento determinado para decidir si existen condiciones no normales de los valores de los sensores en conjunto. Dado que JADE utiliza originalmente una ontología propia para el paso de mensajes, se decidió hacer esta comunicación de forma interoperable a través de servicios Web. Por eso se instaló el addon WISG que transforma esos mensajes en servicios Web. A través de *Web Services Description Language* (*WSDL*) y *Simple Object Access Protocol* (*SOAP*) se pasan los datos al servidor de aplicaciones Web *Tomcat*. Otro motivo por el que usamos *Tomcat* es que estamos interesados en utilizar un navegador Web compatible con la recepción de video en *Hiper Text Markup Language* (*HMTL*) 5.

Si se genera una alarma se dispara la *PiCam* para registrar lo que ocurre en el Hogar. Esta cámara se conecta al servidor de video Streaming propio de la RaspBerry Pi B+ llamado *RPiCam* que no registra audio de forma nativa pero si video en formato *Motion Joint Photographic Expert Group* (*MJPEG*) versión 4. El cual se comunica en formato H.264 para que se pueda recibir en HTML5 en el navegador Web (que puede ser cualquiera que acepte video en HTML5: prácticamene cualquier navegador actual).

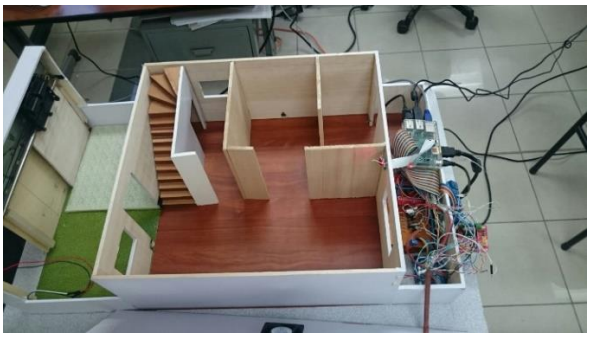

Fig. 5. Fotografía de la maqueta construida para emular situaciones reales de intrusiones no deseadas en un Hogar.

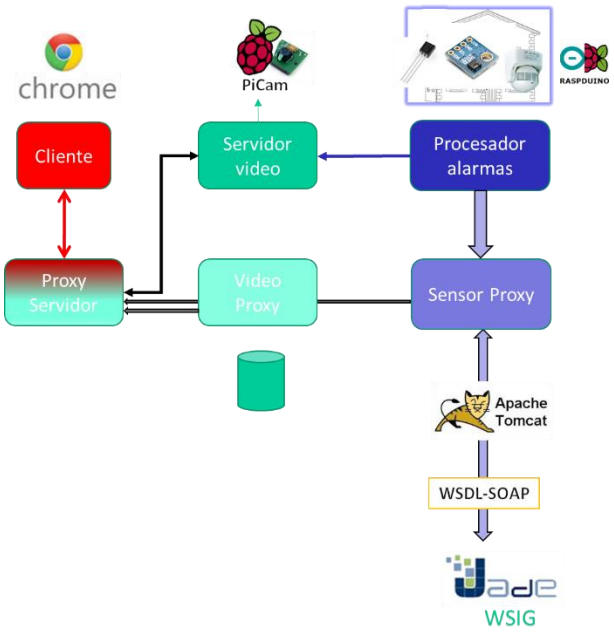

Fig. 6. Esquema del uso del software libre para la implantación de los componentes del sistema de videovigilancia.

C. Resultados experimentales de la mitigación de interrupciones de video streaming y la QoE

Para verificar el funcionamiento del sistema de videovigilancia se hicieron múltiples pruebas. En ellas dos aspectos importantes a observar es el comportamiento ante diferentes tipos de interrupciones de video streaming y la medida subjetiva de la QoE.

Se hicieron pruebas de tiempo de ejecución en presencia de interrupciones del video streaming que se provocaban moviendo un teléfono móvil de alta gama fuera y dentro del área de cobertura de un punto de acceso WiFi, al objeto de medir el tiempo total que se tardaba en emitir un video de 300 s. En los casos en los que las interrupciones duraban poco tiempo y eran sólo 3 interrupciones, era aceptable continuar con la visualización del video. Pero hubo casos en los que cerca del 50% del tiempo de visualización se invirtió en interrupciones, con lo cual era inviable estar cerca 400 s para ver el video entero.

Para medir la QoE se hicieron pruebas de MOS. Se determinó el tamaño de la muestra en consonancia con [24] (5% de error, con un nivel de confianza del 95% y una heterogenidad del 50%), obteniendo un total de 25 personas. Para la escala de opinión se utilizó la norma *International Telecommunication Union* (*ITU*) *P.800.1*, y se hicieron las siguientes preguntas (usando un formulario de *Google Forms*): *¿cómo calificaría los siguientes parámetros de la aplicación*? (*diseño de la interfaz, usabilidad, interactividad*) *¿tuvo problemas en el acceso a la aplicación*? *¿la aplicación tuvo algún error de ejecución*? *¿la aplicación reconectó el vídeo sin recargar la página*? *¿la aplicación envió las alertas de seguridad*? Las clases de respuestas para la primera pregunta fueron: *Excelente*, *Muy bueno*, *Bueno*, *Mejorable*, *Muy mejorable*. Y para el resto de preguntas *Si* y *No*. En la Fig. 7 se muestran las respuestas para la primera pregunta destacando el elevado grado de satisfacción del usuario. Las dos últimas preguntas tuviero un 100% de respuestas *Si*. El resto de preguntas tuvo un valor superior al 90% de respuestas *No*.

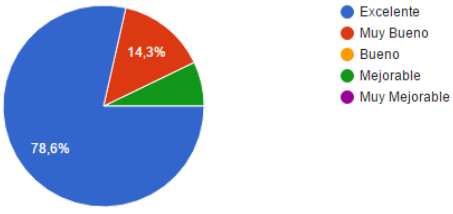

Fig. 7. Respuestas proporcionadas para la pregunta ¿Cómo calificaría los siguientes parámetros de la aplicación? (diseño dela interfaz)

V. CONCLUSIONES

Los sistemas de videovigilancia del Hogar son muy comunes a día de hoy. Con el auge de la IoT y los computadores embebidos en una placa de muy bajo coste económico, y el software libre es posible implantar sistemas de videovigilancia de muy bajo coste capaces de mitigar los efectos adversos de las interrumpciones de video en el cliente. Nosotros mostramos que es posible modelar el software con patrones de diseño y ontologías para el diseño de la inteligencia del sistema de control de alarmas. También mostramos que a través de figuras de medida como el MOS se satisface ampliamente la QoE del usuario en un porcentaje muy elevado.

Como trabajo futuro planteamos el diseño de un sistema proactivo de estimación inligente de las interrupciones de video que permita adelantar información al usuario de las interrupciones que podría experimentar.

AGRADECIMIENTOS

This work has been funded by the Spanish Ministry of Economy and Competitiveness/FEDER under project TEC2015-67387- C4-4-R. Agradecer a las personas que se ofrecieron a hacer las encuestas para el MOS.

REFERENCIAS

[1] Ministerio del Interior de España, *Infracciones Penales Registradas En Ccaas, Provincias, Islas, Capitales y Localidades con Población Superior A 50.000 Habitantes*. Disponible en: http://www.interior.gob.es/documents/10180/5791067/informe+balance+2016_ENE_MARZO.pdf/2a4b89ea-6ddc-448f-870a-428412cbf691.

[2] CEDATOS, *Los problemas que más preocupan a los ecuatorianos después del terremoto*. Disponible en: https://www.cedatos.com.ec/detalles_noticia.php?Id=259.

[3] Agencia española de protección de datos, *Videovigilancia*. Legislacción disponible en: http://www.agpd.es/portalwebAGPD/canaldocumentacion/informes_juridicos/videovigilancia/index-ides-idphp.php.

[4] Kumar Mandula; Ramu Parupalli; CH.A.S. Murty; E. Magesh; Rutul Lunagariya, *Mobile based home automation using Internet of Things (IoT)*, 2015 International Conference on Control, Instrumentation, Communication and Computational Technologies (ICCICCT), pp: 340-343, 2015.

[5] Sagar R N, Sharmila S P, Suma B V, *Smart Home Intruder Detection System*, International Journal of Advanced Research in Computer Engineering & Technology (IJARCET) Volume 6, Issue 4, April 2017, ISSN: 2278 – 1323.

[6] Salamone, F.; Belussi, L.; Danza, L.; Galanos, T.; Ghellere, M.; Meroni, I. Design and Development of a Nearable Wireless System to Control Indoor Air Quality and Indoor Lighting Quality. Sensors 2017, 17, 1021.

[7] Sanjana Prasad, P.Mahalakshmi, A.John Clement Sunder,R.Swathi, *Smart Surveillance Monitoring System Using Raspberry PI and PIR Sensor*, International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 5 (6), pp: 7107-7109, 2014, 7107-7109, ISSN: 0975-9646.

[8] Dmitry Stepanov, Igor Tishchenko, *The Concept of Video Surveillance System Based on the Principles of Stereo Vision*, 18th Conference of Open Innovations Association and Seminar on Information Security and Protection of Information Technology (FRUCT-ISPIT), 2016, ISSN: 2305-7254.

[9] Amal Ben Hamida, Mohamed Koubaa, Henri Nicolas, Chokri Ben Amar, *Video surveillance system based on a scalable application-oriented architecture*, Springer Verlag Multimedia Tools and Applications, Volume 75, Issue 24, pp 17187–17213, December 2016.

[10] Wang Zai-Ying, Chen Liu, *Design of Mobile Phone Video Surveillance System for Home Security Based on Embedded System*, 27th Chinese Control and Decision Conference (CCDC), pp: 5856-5859, 2015.

[11] Elsa Mª Macías and Alvaro Suarez, *Video Streaming based Services over 4G Networks: Challenges and Solutions*, Book Chapter: Fourth-Generation Wireless Networks: Applications and Innovations, Adibi, Sasan, Amin Mobasher and Mostafa Tofighbakhsh (editores), Chapter 22, pp: 494 - 523. Estados Unidos de América: IGI Global, 2010. ISBN 978-1-61520-674-2.

[12] Elsa Macías; Alvaro Suárez; F. Espino, *Multi-platform video streaming implementation on mobile terminals*. Alvaro Suarez and Elsa Macias Lopez (editores). "Multimedia Services and Streaming for Mobile Devices: Challenges and Innovations." 1-350 (2012), accessed December 18, 2012, doi:10.4018/978-1-61350-144-3, Capítulo 14, pp. 288 – 314, Estados Unidos de América: IGI Global, 2012, ISBN: 978-1-61350-144-3.

[13] F. Bellifemine, A. Poggi, and G. Rimassa, "JADE–A FIPA-compliant agent framework," in *Proceedings of PAAM*, 1999, vol. 99, no. 97–108, p. 33.

[14] D. Greenwood, P. Buhler, and A. Reitbauer, "Web service discovery and composition using the web service integration gateway," in *e-Technology, e-Commerce and e-Service, 2005. EEE'05. Proceedings. The 2005 IEEE International Conference on*, 2005, pp. 789–790.

[15] A. Aloman, A. I. Ispas, P. Ciotirnae, R. Sanchez-Iborra, and M. D. Cano, *Performance evaluation of video streaming using MPEG DASH, RTSP, and RTMP in mobile networks*, in 2015 8th IFIP Wireless and Mobile Networking Conference (WMNC), Oct 2015, pp. 144–151.

[16] Laksono, I., *Achieve End-To-End Qos for Wireless Video Streaming, 2004*. Disponible en: http://www.eetimes.com/document.asp?doc_id=1272006.

[17] Navin Sharma, Dilip Kumar Krishnappa, David Irwin, Michael Zink, and Prashant Shenoy, *GreenCache: Augmenting Off-the-Grid Cellular Towers with Multimedia Caches*, Proceedings of the 4th ACM Multimedia Systems Conference, 271-280, 2013.

[18] Mohammad Abu-Lebdeh, Fatna Belqasmi, and Roch Glitho, *An Architecture for QoS-Enabled Mobile Video Surveillance Applications in a 4G EPC and M2M Environment*, IEEE Access, Volume 4, pp: 4082-4093, August 2016, Digital Object Identifier 10.1109/ACCESS.2016.2592919.

[19] Tatiana Gualotuña, *Diseño de una Platalforma de Agentes para Control de Servicios de Video Streaming Móvil*, PhD Thesis, Supervisors: Alvaro Suárez y Elsa Macías, Universidad de Las Palmas de Gran Canaria (ULPGC), España, Febrero 2016.

[20] Andrés Rivandeneira, *Video Vigilancia Autonoma mediante Sistemas Empotrados-Hardware Libre*, Master Thesis, Supervisora: Tatiana Gualotuña, Universidad de Las Fuerzas Armadas (ESPE), Ecuador, 2016.

[21] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, *Design patterns: elements of reusable object-oriented software*, Addison-Wesley, 1995, ISBN: 0-201-63361-2.

[22] Brian Henderson-Sellers, Cesar Gonzalez-Perez, Tom McBride, Graham Low, *An ontology for ISO software engineering standards: 1) Creating the infrastructure*, Elsevier Computer Standards & Interfaces, 36 (2014) 563–576.

[23] C. Gonzalez-Perez, B. Henderson-Sellers, T. McBride, G.C. Low, X. Larrucea, *An Ontology for ISO software engineering standards: 2) Proof of concept and application*, Computer Standards & Interfaces 48 (2016) 112–123.

[24] Takuto Kimura, Masahiro Yokota, Arifumi Matsumoto, Kei Takeshita, Taichi Kawano, Kazumichi Sato, Hiroshi Yamamoto, Takanori Hayashi, Kohei Shiomoto, and Kenichi Miyazaki, *QUVE: QoE Maximizing Framework for Video-Streaming*, IEEE Journal of Selected Topics in Signal Processing, Vol. 11, No. 1, February 2017.