UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

DEPARTAMENTO DE INFORMÁTICA Y SISTEMAS

TESIS DOCTORAL

AGENTES SOFTWARE EMOCIONALES PARA LA RECUPERACIÓN AUTOMÁTICA DE SESIONES DE VIDEO-STREAMING CON DISPOSITIVOS MÓVILES

Francisco José Espino Espino

Las Palmas de Gran Canaria, 2015

Departamento: **Informática y Sistemas**

Programa de doctorado: **Tecnologías de la Información y sus Aplicaciones**

Título de la Tesis

Agentes Software Emocionales para la Recuperación Automática de Sesiones de Video-Streaming con Dispositivos Móviles

Tesis Doctoral presentada por D. **Francisco José Espino Espino**

Directores: Dr. D. **Álvaro Suárez Sarmiento**
Dra. Dª. **Elsa María Macías López**

Los directores, **El doctorando,**

Las Palmas de Gran Canaria, a 30 de octubre de 2015

A mis padres.

Agradecimientos

Esta tesis ha sido toda una aventura. Una de esas con final incierto y certeza de fracaso hasta el último momento. Una arriesgada acrobacia que con ambición e independencia ha acabado bien pese a la gran dificultad, a la enorme resistencia institucional y al elevado coste económico y personal que ha supuesto. Ciertamente "audentis Fortuna iuvat".

Muchas personas han aportado algo de una u otra manera en este trabajo y por ello estoy muy agradecido. Destacar a mis padres Manolo y Paqui, y a mi hermana Elisenda, por acompañarme siempre, animarme y apoyarme incondicionalmente. A mis directores de tesis Álvaro y Elsa por guiarme y ayudarme en esta complicada labor. Y como no, a una infinidad de colegas, amigos y amigas que tengo la suerte de conocer, por interesarse y compartir conmigo su vasto conocimiento en multitud de materias y disciplinas científicas, filosóficas, técnicas y artísticas, siendo una fuente valiosísima e inspiradora para el desarrollo y conclusión de esta tesis.

Finalmente quiero mencionar también a todas esas personas más o menos anónimas que con su esfuerzo y dedicación han apostado y apuestan por liberar la cultura y hacerla más accesible a todo el mundo. Además de promover una sociedad mucho mejor, facilitan enormemente la labor de los creadores en general y, por ende, de investigaciones como ésta.

Resumen

Los dispositivos móviles han evolucionado en pocos años hasta lograr prestar una gran cantidad de servicios. Aquellos que requieren comunicaciones con otros dispositivos se enfrentan a la problemática inherente del empleo de redes inalámbricas: comportamiento caótico y desconexiones frecuentes. Con el *video-streaming* esta circunstancia es una dificultad considerable.

Ante la presencia de desconexiones en la red el *video-streaming* puede desde interrumpirse temporalmente hasta quedar invalidado permanentemente perdiendo la sesión establecida. En este trabajo se propone una solución basada en agentes software emocionales con la que minimizar los efectos negativos de esta situación.

Estos agentes software emocionales son de inspiración biológica y poseen distintas habilidades cognitivas. Entre ellas la principal es la capacidad de predecir el futuro inmediato en un entorno caótico con una tasa de acierto elevada. En este caso el entorno es la red inalámbrica y en ella los agentes consiguen actuar antes de que se produzcan momentos críticos capaces de interrumpir el *video-streaming* maximizando de este modo la posibilidad de que transcurra con éxito.

Para emular capacidades mentales artificialmente se hace uso de un modelo descriptivo de la combinación intuición-pensamiento que aunque hipotético resulta coherente con el conocimiento disponible actualmente en neurociencia cognitiva. Para justificar su eficacia se incluyen diferentes pruebas.

Índice general

Índice de figuras

Capítulo 1

Introducción

Resumen: este capítulo trata del contexto en el que se sitúa el trabajo de investigación realizado, su motivación y los objetivos perseguidos. También incluye indicaciones sobre la estructura de la presente memoria.

1.1 Consideraciones iniciales

Un dispositivo móvil se caracteriza esencialmente por ser un computador de uso personal que por su forma, tamaño y peso puede llevarse en una mano mientras se utiliza. Esto es una "máquina electrónica dotada de una memoria de gran capacidad y de métodos de tratamiento de la información, capaz de resolver problemas aritméticos y lógicos gracias a la utilización automática de programas registrados en ella" [1]. El grupo de los computadores de uso personal e individual puede dividirse en varias categorías según el grado de movilidad que presentan los mismos, determinado éste principalmente por el peso, el tamaño y la independencia energética de la máquina. Los más grandes son los computadores de sobremesa que están diseñados para mantener una ubicación fija mientras son utilizados. Al no tener restricciones energéticas ni de espacio físico pueden incorporar una variedad mayor de componentes, optando generalmente por los de más rendimiento. A continuación se encuentran los computadores portátiles, mucho más pequeños y

ligeros, que pueden transportarse con facilidad. Tienen una cierta autonomía al poder operar un período de tiempo sin estar conectados a la red eléctrica pero siempre en un entorno estático, sin movimiento de la máquina. Funcionalmente los computadores de sobremesa y portátiles son muy parecidos y pueden compartir programas informáticos entre ellos. Finalmente la última categoría de computadores personales, los más pequeños e independientes, la forman los dispositivos móviles. Están pensados para ser manejables, operar de forma completamente autónoma y resistir movimientos físicos durante su uso. Funcionalmente son muy diferentes de los de mayor tamaño al estar mucho más limitados en recursos como memoria, pantalla o capacidad de cálculo, y por ello requieren de programas informáticos especialmente diseñados para ellos [2]. Sin embargo, estos dispositivos suelen tener una gran capacidad de comunicación usando distintas tecnologías y un número elevado de sensores.

El uso de los dispositivos móviles se ha generalizado en los últimos años. Estas máquinas proporcionan servicios muy específicos en los que el factor movilidad les adjudica un valor añadido considerable, poniéndolas en esos casos por delante de las de mayor tamaño, los computadores de propósito general (de sobremesa y portátiles). La especialización de los dispositivos móviles hace que tengan la capacidad de cálculo y memoria justas para su cometido, siendo generalmente escasas en general. Con ello consumen menos energía mejorando la autonomía que le proporciona las baterías de las que se alimentan y les hace independientes de la red eléctrica cuando son utilizados. Sus sistemas operativos también están especialmente diseñados para ahorrar consumiendo pocos recursos y optimizados para trabajar con eventos externos, situándose en cuanto a la gestión de los mismos a lo largo del tiempo entre los laxos sistemas operativos convencionales y los estrictos sistemas operativos de tiempo real. Otra

característica básica de los dispositivos móviles es que pueden conectarse y comunicarse con otros dispositivos ya sea por cable o inalámbricamente ampliando así sus capacidades iniciales.

Para clasificar un dispositivo móvil entre los diferentes tipos existentes en el mercado hay que considerar la fisonomía que posee, los servicios que puede prestar y el rendimiento que alcanza en cada uno de ellos, y considerar que las diferentes categorías que se pueden dar como, entre otras, teléfonos móviles, tabletas, videoconsolas portátiles, *Asistentes Personales Digitales* (PDA), geolocalizadores, reproductores multimedia, cámaras digitales de fotos o vídeo, no son categorías disjuntas: por ejemplo existen cámaras fotográficas que proporcionan servicios de geolocalización o tabletas de reducido tamaño que hacen fotografías y llamadas telefónicas. Una nueva categoría que ha surgido de la combinación principalmente de un teléfono móvil con una PDA es la de teléfono móvil inteligente, o *smartphone,* constituyendo uno de los grupos de dispositivos móviles más relevantes en nuestros días.

Los teléfonos móviles inteligentes son un tipo de dispositivo móvil muy común y realmente numeroso, superando actualmente los mil millones de aparatos en todo el mundo [3] y con perspectivas de importante aumento de su población en el futuro. Este éxito se debe principalmente al carácter práctico de estos dispositivos. Su tamaño físico es más bien reducido, caben en la palma de una mano, y son extremadamente polivalentes, pueden realizar un gran número de tareas diferentes como grabar y reproducir contenidos multimedia (audio y vídeo), navegar por *Internet* [4], usar servicios de geolocalización basado en el *Global Positioning System* (*GPS*) [5], ejecutar aplicaciones móviles complejas, de asistencia personal o videojuegos, y servir de terminal de telefonía móvil como su propio nombre indica. Un caso particular es la reproducción de contenidos multimedia generados o

ubicados en un equipo externo o servidor y que son comunicados por la red. Existen varios métodos para ello siendo los más interesantes los que permiten simultanear la descarga con la reproducción del contenido multimedia. Esto se conoce como *streaming* [6] y su utilización y problemática en dispositivos móviles, sobre todo en teléfonos móviles inteligentes, son el fundamento de este trabajo, con especial énfasis en el *streaming* de vídeo el cual acapara actualmente la mayor parte del tráfico de datos de estos dispositivos y se prevé que sea el que más crezca en los próximos años [7].

Antes de la aparición del *streaming* se reproducía un vídeo ubicado en un servidor primero descargándolo completamente en el cliente y después iniciando su reproducción. En este caso las tareas necesarias de descarga y visualización se realizan secuencialmente. Esto tiene dos inconvenientes fundamentales acrecentados por su cantidad de información: elevado número de fotogramas y *pixels* por fotograma (peso o tamaño del vídeo) que suelen tener los vídeos aunque tengan una duración relativamente pequeña. El primero de estos inconvenientes es el tiempo de espera necesario para que se realice la descarga, normalmente largo, y el segundo el espacio necesario para guardar el contenido descargado, generalmente bastante grande. El *streaming* soluciona en gran medida estos dos problemas de manera satisfactoria.

La técnica de *streaming* aprovecha que el vídeo se divide en paquetes del mismo tamaño que suelen agrupar a varios fotogramas o *frames* de vídeo. Los distintos paquetes se envían ordenadamente desde el primero hasta el último en secuencia desde el servidor de vídeo hasta el cliente reproductor o visualizador. Solapadamente en el tiempo, a la vez que el servidor de vídeo está enviando un paquete, por la Red pueden viajar otros y el cliente puede mostrar otros. Con ello se consigue ejecutar en paralelo la emisión por parte del servidor, la comunicación

por la red y la recepción y reproducción en el cliente. En este caso el tiempo de espera para empezar a reproducir corresponde al retraso de recepción del primer paquete. El espacio de memoria necesario es también mínimo porque estos paquetes una vez reproducidos se eliminan de la memoria. La inmediatez y la escasa utilización de recursos del *streaming* lo hacen muy apropiado y conveniente para ser empleado en teléfonos móviles inteligentes, en los que la eficiencia se hace incluso más necesaria que en los computadores de propósito general, ya que los teléfonos móviles inteligentes disponen de menos recursos. Sin embargo, la técnica de *streaming* requiere de un canal de comunicación en condiciones suficientes de ancho de banda y estabilidad para el envío fluido y monótono de los paquetes. Los teléfonos móviles inteligentes modernos disponen de tecnologías de comunicación de banda ancha, como por ejemplo *Wireless Fidelity* (*WiFi*) [8] o *High Speed Downlink Packet Access* (*HSDPA*) [9] o *Long Term Evolution* (*LTE*) [10], que permiten el uso de *streaming* al lograr una velocidad de comunicación aceptable que permite ocultar en todo el proceso la latencia propia de la red. Sin embargo, estas tecnologías se basan en canales de comunicación inalámbricos que no pueden garantizar a lo largo del tiempo una estabilidad suficiente en su funcionalidad.

El comportamiento caótico espacio-temporal de los canales inalámbricos [11] provoca variaciones de ancho de banda frecuentes y en el peor de los casos hasta desconexiones esporádicas de los dispositivos invalidando completamente el canal. Estas desconexiones tienen causas muy variadas y pueden producirse a diversos niveles de la arquitectura de red. Todo esto hace que el *streaming* en teléfonos móviles inteligentes pierda efectividad y a veces sea impracticable. Una desconexión de este tipo supone para el usuario la pérdida de la sesión de *streaming* y la obligación de iniciar otra sesión para reanudar la reproducción del contenido, lo cual supone un tiempo, un volumen de datos y un esfuerzo que pueden ser

inasumibles en muchas ocasiones. También pueden darse casos como por ejemplo en la transmisión de un vídeo en vivo (de generación y transmisión simultánea) en el que la porción emitida durante la desconexión se pierde irremediablemente y no puede ser recuperada posteriormente, lo cual va en perjuicio de la experiencia del usuario. Para el servidor y la red una desconexión también es un problema. Al servidor se le obliga a realizar movilizaciones y desmovilizaciones continuas de recursos que son costosas, e innecesarias si no se produjeran estas desconexiones. También hay que tener en cuenta que en el intervalo de tiempo que hay entre la desconexión y su detección se transmiten paquetes que no van a ser consumidos saturando la red de forma inútil.

Pero las desconexiones, tan corrientes en las redes inalámbricas, son sólo parte del problema. Hay que considerar que para mantener una sesión de *streaming* debe garantizarse un mínimo de ancho de banda en la red utilizada, dependiente éste de las características del contenido transmitido. Cualquier disminución del ancho de banda en la que quede por debajo de ese límite puede desde añadir pequeños retrasos en el flujo de datos hasta bloquearlo temporal o permanentemente afectando directamente a la reproducción del contenido. El caso más radical de disminución de ancho de banda es el de una desconexión en la que éste queda anulado y el flujo de datos se bloquea permanentemente. Aunque no es posible aumentar el ancho de banda de una red sin modificarla o subsanar una desconexión que ha sido inevitable sí que se pueden tomar ciertas medidas para minimizar los daños producidos al perder el flujo de datos. En el caso del *streaming* con teléfonos móviles inteligentes estas acciones se vuelven imprescindibles, mucho más que en las redes por cable donde el ancho de banda no fluctúa tanto y tan rápido y las desconexiones son más bien inusuales y excepcionales, todo lo contrario que en las redes inalámbricas.

Las acciones que se pueden llevar a cabo para mejorar la calidad del *streaming* en redes inalámbricas se ven limitadas por la elevada frecuencia de variación del ancho de banda efectivo del canal sin causas claramente motivadas. Estas acciones pueden ser reactivas y ejecutarse tras la ocurrencia de un evento problemático. El intervalo de tiempo entre el evento causante y la acción consecuente, determinado éste por el tiempo necesario para la detección del evento, hace que algunas veces esta solución sea ineficaz por la lentitud de respuesta asociada, lo que puede añadir retrasos o incluso pérdida de datos en la comunicación. La solución ideal en este caso tiene que permitir anticiparse a los eventos problemáticos que se pueden dar. Sin embargo, realizar predicciones del comportamiento de un canal inalámbrico con un elevado grado de acierto, condición necesaria para la citada solución, es muy complicado y en la mayor parte de las veces imposible con los medios actuales. Esto se debe a que el entorno de una red inalámbrica es el medio físico real por donde viajan las ondas electromagnéticas y éste es muy parcialmente observable, no es posible conocer en su totalidad el estado de todos sus componentes. Este es un ejemplo de sistema no-lineal y caótico [12] en el que existen muchas variables relevantes, más de las observables, que son continuas, muy dinámicas y que cambian frecuentemente de forma irregular, sin un patrón claro y conocido de antemano. En esta situación conseguir predicciones concretas y acertadas con las que responder de forma proactiva a las dificultades que se pueden presentar en las comunicaciones inalámbricas no es técnicamente viable en nuestros días. Seguir avanzando en la resolución de este problema es el propósito fundamental de este trabajo y para ello se combinan las herramientas usuales que se aplican en los sistemas dinámicos no-lineales, como el análisis de series temporales [13], con áreas como el aprendizaje automático [14], la neurociencia [15] o la psicología de la emoción [16].

1.2 Motivación

El *streaming* en dispositivos móviles adolece de graves problemas, principalmente por la naturaleza de sus sistemas de comunicación inalámbricos. En primer lugar el comportamiento de un canal inalámbrico (no guiado) es prácticamente impredecible en comparación con el comportamiento de un canal guiado. Se pueden producir aumentos y disminuciones del ancho de banda llegando a puntos de ausencia de señal teniendo como consecuencia la desconexión de los terminales, todo ello en periodos de tiempo cortos. Este panorama para una tarea en tiempo real y de alta carga computacional como el *streaming* requiere de sistemas que como mínimo mitiguen estos problemas.

Los servidores actuales de video-streaming proporcionan medios muy primitivos (poco resolutivos) especialmente orientados a la solución de los problemas típicos que se originan de su utilización con terminales móviles. Las pérdidas de información y la necesidad de establecer una nueva sesión de *streaming* cada vez que se pierde la cobertura son los principales a tener en cuenta, y su tratamiento es de vital importancia para un aumento de la *Calidad de Experiencia* (*QoE*) que experimenta el usuario al reproducir contenidos multimedia en los dispositivos móviles.

Al ser software cerrado, la mayoría de los servidores y clientes de video-streaming con posibilidad de ser usados en dispositivos móviles se hace necesario el uso de componentes intermediarios independientes para añadir nuevas funcionalidades. Éstos pueden tomar forma de monitores de red controlando ciertos parámetros del tráfico de datos e interactuando con el servidor y el cliente, o también pueden actuar como representantes (*proxies*) del servidor o del cliente ampliando sus capacidades.

Otro aspecto relevante en el desarrollo de una solución a estos problemas reside en la propia naturaleza de los dispositivos móviles. Aunque hayan aumentado en gran medida en capacidad, sus recursos siguen siendo escasos. Por lo tanto se requieren aplicaciones livianas. Otro condicionante es la enorme heterogeneidad de la población de estos aparatos, por lo que para llegar a un número mayor de los mismos hay que basar las soluciones en tecnologías y herramientas de desarrollo con gran difusión entre ellos.

En general para controlar la inestabilidad propia de las redes inalámbricas se han utilizado *proxies* con los que adaptar la velocidad de transmisión, teniendo en cuenta el estado del canal inalámbrico [17], de los recursos previamente almacenados [18], o del vídeo en vivo [19]. Estos *proxies* se han programado como agentes software de una plataforma propietaria [20] para resolver las desconexiones mientras se resuelven los procesos de itinerancia (*roaming*) usando un esquema de memorización intermedia (*buffering*) proactivo; sin embargo, no se ha aplicado en la práctica sobre arquitecturas de teléfonos móviles comerciales.

La incorporación de una memoria intermedia o *buffer* en un canal inalámbrico utilizado para realizar *streaming* permite amortiguar las variaciones del retraso de transmisión de paquetes (*jitter*) y los efectos de la congestión de la red [21]. Todo ello a costa de introducir un retraso constante en todos los paquetes el cual debe ser tomado en consideración ya que aunque indeseado es proporcional a la capacidad de amortiguación del *buffer*. Con poco retraso, propio de un *buffer* de poca capacidad, se consigue también poca amortiguación y con mucha amortiguación, propia de un *buffer* grande, el retraso aumenta. Otro problema resulta de la dificultad de controlar un *buffer* de estas características, lo cual no es trivial y mucho menos en un canal inalámbrico [22] con dispositivos móviles que tienen requisitos especiales de eficiencia y bajo consumo de energía [23]. Un *buffer*

se comporta como una cola *First In First Out* (FIFO). Para que no se corte el flujo de datos el buffer no puede ni vaciarse ni llenarse por completo. En *streaming* se siguen diferentes estrategias con este fin. Para que el buffer no se vacíe hay que tener en cuenta si se puede modificar la velocidad de transmisión [24] o no [25]. En el caso de que se llene se pueden borrar paquetes seleccionados según dependencias que mantengan entre ellos [26]. Aunque generalmente estas acciones se realizan en base a reglas predefinidas también se ha experimentado con algoritmos más complejos como las redes neuronales o la lógica difusa [27] con el fin de ejercer un comportamiento adaptado a las circunstancias del entorno aunque no se ha avanzado mucho por este camino.

En *streaming* la utilización de *proxies* con memorias intermedias más allá de lo estrictamente necesario resulta negativo. Añaden retrasos y consumen muchos recursos al realizar una tarea tan exigente en cuanto a cantidad de datos procesados y requisitos temporales. Resulta más productivo utilizar componentes intermedios que no interfieran en el flujo de datos pero que sean capaces de monitorizarlo y tomar las decisiones pertinentes para adaptar la velocidad de transmisión a las circunstancias del canal inalámbrico no re-procesando los datos como en el caso de los *proxies* con memorias intermedias sino dándole ordenes al servidor para que adecúe su velocidad de transmisión [28]. Sin embargo, existen casos en los que los *buffers* sí que son imprescindibles para que no se produzcan pérdidas de datos: en las desconexiones y en las pérdidas acusadas de la calidad del canal utilizado que impidan el flujo de datos. Éstas pueden ser radicales y durar una cierta cantidad de tiempo o se puede dar la posibilidad de re-conectar inmediatamente a otra estación base o punto de acceso, proceso conocido como handover el cual requiere una gestión específica del buffer [29] para el tráfico general y mucho más estricta para el de streaming por sus inherentes restricciones

temporales. El handover óptimo es aquel que se realiza en el último momento antes de que se pierda la calidad requerida en las comunicaciones. Para ello es necesario actuar justo antes y por lo tanto predecir este instante. Se han utilizado para ello herramientas como redes neuronales [30] o filtros predictores del gradiente de la intensidad de la señal de la red inalámbrica [31]. No obstante considerar sólo la intensidad de la señal o incluso la calidad del contenido recibido es insuficiente [32] para determinar el estado del canal. En base al rendimiento global del streaming también se han desarrollado mecanismos [33] pero éstos se comportan de forma reactiva, no predicen las bajadas de rendimiento como sería ideal.

Otro aspecto a tener en cuenta en el *streaming* con teléfonos móviles inteligentes es el de los protocolos de red usados. Inicialmente éstos fueron diseñados para redes cableadas y se hace necesario una adaptación al nuevo entorno inalámbrico y su problemática [34]. Por un lado existen propuestas para ampliar los protocolos existentes añadiendo una mayor tolerancia a los retrasos y así no asociarlos siempre a la congestión de la red [35] o incorporando conceptos como el de desconexión [36], y por otro propuestas para aplicar técnicas de optimización *cross-layer*, es decir, romper los límites impuestos por los diferentes niveles del modelo de red aprovechando las relaciones entre ellos [37] y así controlar mejor variables como el volumen de información (*throughput*), la tasa de errores, el retraso y su variación (*jitter*), con las que diseñar mejores estrategias de empaquetado y retransmisión [38]. También existen otras alternativas como las redes dinámicas tolerantes a desconexiones las cuales son redes ad hoc donde todos los nodos están al mismo nivel sin necesidad de ninguno central. Estas redes se componen y van tomando forma teniendo en cuenta la frecuencia y duración de las desconexiones de los distintos dispositivos que la forman [39]. La principal dificultad de estas redes reside en el encaminamiento para el cual se han propuesto

varios algoritmos destacando el encaminamiento oportunista [40] pero todavía no se ha llegado a una solución óptima para este problema. Inicialmente este tipo de red ha sido muy interesante para robótica ya que permite una mayor movilidad de los nodos [41] y actualmente también se está usando en *streaming* con dispositivos móviles a nivel experimental [42] quedando su uso a nivel comercial todavía lejano.

En definitiva el objetivo siempre es el mismo: aumentar la QoE del *streaming*. En redes inalámbricas los esfuerzos se han centrado tanto en *Calidad de Servicio* (*QoS*) como en QoE existiendo mucho trabajo realizado al respecto. La QoS en *streaming* se basa esencialmente en que el flujo de datos transcurra sin variaciones ni pérdidas desde que éste se genera hasta que se reproduce. Los problemas más usuales a este nivel además de posibles errores son los retrasos acumulados o su variación (jitter) en la comunicación de paquetes y también la pérdida de los mismos en la red. Todo ello se puede controlar en todos los niveles por separado, ya sea el nivel de control de acceso al medio [43] o el nivel de enlace de datos [44] o el nivel de aplicación [45], o en conjunto mediante cross-layer [46]. La QoS no sólo se aplica a los distintos nodos individualmente sino también a la red en su conjunto evaluando su capacidad para mantener el streaming de cada usuario correctamente [47]. La QoE es un concepto mucho más global y se centra en la aceptabilidad del servicio por parte del usuario. Para evaluarla se suele emplear la opinión de los usuarios la cual es una medida subjetiva difícil de gestionar aunque existen modelos matemáticos para ello [48]. Esto obliga muchas veces a utilizar medidas más objetivas [49] e incluso híbridas [50]. Otra forma de evaluar la QoE es mediante la observación del usuario aprovechando la relación causal entre QoE y comportamiento del usuario [51]. La QoS y la QoE están relacionadas en algunas situaciones pero generalmente no tienen una correspondencia directa y constante entre ellas [52]. Esto no impide que puedan complementarse y usarse

conjuntamente aprovechando toda la información derivada de los distintos niveles de la arquitectura de red. El fin de todo ello ha sido adaptar el tráfico generado por el *streaming* a las cirscunstancias del momento [53] incluso ampliando el margen de acción a la adaptación de la codificación del contenido [54] y a la aplicación de soluciones avanzadas, como por ejemplo en el problema de la congestión de red, al poder decidir qué paquetes pueden ser eliminados de la circulación sin afectar en gran medida a la QoE [55], o en el encaminamiento de redes inalámbricas ad hoc o malladas [56].

El *streaming* en teléfonos móviles inteligentes ha mejorado mucho gracias a los numerosos avances realizados sobre el tema, pero las pérdidas intermitentes de conexión siguen siendo un problema no resuelto de manera satisfactoria con componentes no invasivos que añadan retrasos innecesarios a la comunicación. Que la sesión iniciada por el usuario no se pierda cada vez que la calidad del enlace inalámbrico disminuya y que tampoco se pierda información en el transcurso de la misma es necesario para que el uso de teléfonos móviles inteligentes no vaya en perjuicio del *streaming*. Aumentar la fiabilidad en este caso supone un gran beneficio en el aumento de la calidad, tanto de servicio como de experiencia de usuario. Esto tiene un gran impacto ya que tanto los teléfonos móviles inteligentes como el *streaming* son tecnologías muy difundidas en la actualidad y cuyo empleo seguirá aumentando en el futuro.

1.3 Objetivos

El medio de comunicación característico de los dispositivos móviles es el inalámbrico. Éste no es fiable debido a que su calidad depende mucho del entorno, es decir, de las características del lugar geográfico en el que se ubiquen los

dispositivos móviles. Por lo general éste es imposible de controlar o resulta muy costoso hacerlo. Así que al no poder adaptar el entorno al medio hay que adaptar el medio al entorno.

El control de redes inalámbricas sigue eminentemente estrategias reactivas. Una vez sucedido un problema éste se intenta solucionar. Esto exige una cantidad de tiempo para la detección del problema y la puesta en marcha de la solución. En una actividad tan exigente y de tiempo real como el *streaming* esa espera tiene efectos muy negativos: pérdida de información y sobreutilización de recursos. Esto a su vez se traduce en molestias para el usuario y una disminución de la QoE. En *streaming* la estrategia de control ideal es la proactiva: actuar justo antes de que ocurra un problema. Esto exige predecir con acierto y con antelación suficiente para que dé tiempo de tomar las medidas pertinentes en el momento apropiado.

La predicción del comportamiento de las redes inalámbricas tiene grandes problemas asociados a su carácter caótico y a su baja observabilidad. Aunque el comportamiento caótico de las redes inalámbricas es determinista sigue patrones desconocidos inicialmente que son fruto de la conjunción de todos los factores propios del medio. En la actualidad no existen métodos matemáticos que consigan detectar correctamente estos patrones y utilizarlos para realizar predicciones satisfactoriamente. La creación de un método capaz de ello es el camino para la consecución del objetivo planteado.

No hay que olvidar que la motivación de este trabajo es la recuperación automática de sesiones de *streaming* en dispositivos móviles, con la que minimizar la pérdida de QoE en situaciones críticas. Para que la predicción de interrupciones sea útil y pueda aprovecharse es necesario la implementación de un sistema informático que actúe en base a ella, poniendo en práctica las medidas pertinentes

en cada momento. Además este sistema es el que debe ejecutar la citada predicción integrando el método propuesto.

En resumen, el objetivo básico de este trabajo es el diseño e implantación de un sistema automático para la predicción de interrupciones de *video-streaming* móvil. Esta predicción debe permitir actuar proactivamente ante las interrupciones minimizando al máximo sus efectos negativos.

Para lograr el objetivo marcado se establecen los siguientes subobjetivos o pasos a seguir:

1. Desarrollo de un sistema informático destinado a la recuperación automática de sesiones de *video-streaming* bajo el paradigma de los agentes software.

2. Dotación de cualidades emocionales a los agentes software del sistema desarrollado orientadas éstas a la predicción de interrupciones de *video-streaming*.

3. Validación del sistema desarrollado y análisis de los resultados obtenidos en diferentes pruebas funcionales y de rendimiento.

1.4 Contribuciones

A lo largo de la realización de esta tesis doctoral se ha contribuido a la mitigación del problema de las interrupciones del servicio de *video-streaming* en redes inalámbricas con dispositivos móviles, de la siguiente manera:

- Diseño de métodos matemáticos de predicción basados en emociones:

- Modelo fundamentado en reglas lógicas y proceso de eventos complejos implementado con la herramienta Drools [57].

- Modelo de inspiración biológica más general, flexible y realista emulando la combinación intuición-pensamiento.

- Diseño e implementación de software:

  - Software para distintos teléfonos móviles con los siguientes sistemas operativos: Symbian S60 3.1 [58] (Nokia N95 [59]), Android 2.3.6 Gingerbread [60] (Samsung Galaxy Ace GT-S5830 [61]), Android 4.1.2 Jelly Bean [62] (Samsung Galaxy Ace 2 GT-I8160P [63]), Android 4.4.4 KitKat [64] (Samsung Galaxy Ace 4 SM-G357FZ [65]) y Android 5.1.1 Lollipop [66] (Motorola Moto G (3rd Gen.) [67]).

  - Software para distintos servidores basados en Microsoft Windows [68], Ubuntu Linux [69], Mac OS X [70].

  - Software para distintos tipos de proxies, nativo escrito en C [71] y C++ [72] y multiplataforma escrito en Java [73] y Python [74].

Parte del trabajo anterior se ha desarrollado como colaborando en el proyecto de investigación *Video Streaming Proxy Agents* (*VPA*), Ministerio de Industria, Turismo y Comercio, FIT-330210-2006-54, 2006 y 2007, como miembro contratado por la ULPGC y en colaboración con la empresa INERZA S.A.

Parte del trabajo realizado se ha publicado en congresos, revistas y capítulos de libro internacionales:

- A. Suárez, F. Espino, E.M. Macías, "Uso de JADE-LEAP para recuperar automáticamente sesiones de vídeo-streaming en teléfonos móviles", en las XVIII Jornadas TELECOM I+D, 2008 [75].

- A. Suárez, F. Espino, E.M. Macías, "Automatic recovering of RTSP sessions in mobile telephones using JADE-LEAP", en la revista IEEE Latin America Transactions, 2009 [76].

- E. Macías, A. Suárez, F. Espino, "Multi-platform Video Streaming Implementation on Mobile Terminals", en el libro A. Suárez, E. Macías (Eds.) Multimedia Services and Streaming for Mobile Devices: Challenges and Innovations, Information Science Reference, 2012 [77].

1.5 Organización de la memoria

La organización de este documento está orientada a facilitar la comprensión de todo el proceso seguido para la consecución de los objetivos planteados en 1.3.

En primer lugar se introduce la técnica de *video-streaming*. Se analiza la problemática generada por su utilización con dispositivos móviles y se propone una solución bajo el paradigma de los agentes software.A continuación se relata cómo dotar a agentes software de cualidades emocionales útiles para interactuar en entornos caóticos deterministas parcialmente observables y sobre todo para conseguir predicciones efectivas bajo un horizonte de unos pocos segundos.

Finalmente se enuncian los resultados obtenidos en diferentes situaciones con y sin interrupciones. Éstos son fruto de la previa validación de todo el sistema informático desarrollado incorporando los mecanismos propuestos. El análisis de

estos resultados se realiza a nivel funcional y de rendimiento para evaluar la eficacia y la eficiencia de la solución desarrollada.

Esta memoria termina con las conclusiones a las que se ha llegado y con la propuesta de líneas futuras de trabajo a seguir en base a los resultados conseguidos y al conocimiento aportado.

Capítulo 2

Video-streaming y dispositivos móviles

Resumen: este capítulo incluye los aspectos básicos del video-streaming y la problemática que surge al emplearlo en dispositivos móviles como tabletas o teléfonos móviles inteligentes. Se propone una solución para mitigar los efectos negativos de las interrupciones en la reproducción del *video-streaming*.

2.1 Introducción

La comunicación y difusión de vídeo digital es una tarea problemática debido principalmente a:

1. El vídeo en general ocupa mucho espacio en memoria, lo que ha motivado el uso de diferentes técnicas de *COmpresión-DEsCompresión* (*CODEC*) que sólo mitigan la problemática.

2. Las técnicas de CODEC requieren una elevada capacidad de cálculo para lo que es necesario hardware muy potente.

3. La técnica de DEC en el reproductor se debe realizar en tiempo real firme (*firm real time*) lo que obliga a mejorar considerablemente el software y el hardware que lo implante.

A pesar de ello, la reproducción de vídeo almacenado o emitido en directo desde un servidor remoto de Internet, incluso en enlaces con escaso ancho de banda, es una realidad hoy en día. Para ello se utilizan técnicas de comunicación complejas de implantar que aprovechan al máximo el ancho de banda y la capacidad de cálculo de servidores y reproductores; todo ello acoplado y sintonizado adecuadamente. A continuación se analiza la evolución de estas técnicas.

Originalmente los contenidos multimedia ubicados en servidores conectados a Internet tenían que ser descargados completamente a un medio de almacenamiento local para su reproducción. A nivel de usuario esto es muy poco atractivo ya que el tiempo transcurrido desde la solicitud del audio o vídeo de interés hasta su reproducción puede llegar a ser inaceptable en algunos casos y en otros, cuanto menos, molesto. Otra traba importante para el usuario es la necesidad de espacio físico para almacenar el contenido multimedia requerido. En muchos casos el dispositivo reproductor no tiene la capacidad suficiente.

Para solventar en parte esta situación aparece la *descarga progresiva*. Con ella sigue siendo necesario disponer de gran cantidad de espacio de almacenamiento en un medio local pero el contenito multimedia puede reproducirse durante su descarga tras un tiempo de espera inicial muy bajo. La reproducción puede comenzar desde que se dispone de información suficiente en el medio local donde se guarda la descarga. La comunicación se realiza generalmente a la máxima velocidad posible pudiéndose dar el caso de que la descarga finalice mucho antes que la reproducción. Esto permite soportar pequeñas interrupciones en la descarga, si no se llega a consumir todos los datos disponibles en el reproductor antes de que se reanude, dando cierta robustez al proceso.

La *descarga progresiva* tiene algunos problemas prácticos. Además de necesitar gran cantidad de espacio en una memoria local para la propia descarga, tiende a saturar la red. Otro inconveniente es que el retraso que existe entre la petición de descarga y el comienzo de la reproducción se mantiene en el tiempo y es demasiado grande para lograr comunicaciones en directo. Con la aparición del *streaming* se superaron estos problemas: no se descarga el contenido multimedia, la comunicación se realiza a una velocidad adecuada para la reproducción y el retraso es mínimo. El *streaming* es una técnica de comunicación de datos que permite procesar un flujo secuencial de información de forma casi inmediata y continua, según se vaya recibiendo. Este flujo consiste en uno o varios medios multiplexados, provenientes de uno o varios servidores, que son recibidos en tiempo real por un cliente usando una red. La información se refiere al contenido multimedia que previamente ha sido comprimido en un formato compatible con esta tecnología tal que pueda ser subdividido en paquetes de información transmisibles de forma ordenada al destinatario. Estos paquetes son almacenados en un pequeño espacio de memoria, o buffer, al llegar al cliente. Una vez lleno se comienza la reproducción. Este proceso dura muy poco tiempo después de iniciado el flujo. Los paquetes sólo se guardan el tiempo necesario para ser reproducidos después del cual el lugar que ocupan en el buffer es usado por los nuevos paquetes que van llegando sucesivamente. Al final se vacía todo el buffer. De esta manera se consigue una reproducción fluida y que no se tenga que almacenar ningún archivo.

El *streaming* ha incrementado de forma notable las posibilidades de comunicar gran cantidad de contenidos multimedia en Internet. Esta técnica permite publicar vídeo o audio que el usuario puede reproducir con un tiempo de espera mínimo y sin necesidad de descargar previamente todo el contenido. El usuario inicia una sesión de *streaming* y elige quedarse o abandonar sin mayor

problema. Además, esta técnica aprovecha las redes de comunicación actuales sin necesidad de modificaciones y sin interferir con otras aplicaciones.

El *streaming* presenta algunos inconvenientes. Uno de ellos es que requiere de un ancho de banda efectivo en la Red que en ningún caso puede ser inferior a la tasa de transferencia propia del contenido en cada momento. Si se diera el caso se pueden producir interrupciones en la reproducción. Otro inconveniente es la complejidad de los procesos dedicados al streaming: se requiere de mucha pericia y experiencia para lograr un producto eficiente y eficaz.

Las aplicaciones del *streaming* son muy numerosas. Una de las más destacadas es la posibilidad de transmitir radio y televisión en directo por Internet, cosa impensable hasta la aparición de esta técnica. Las videoconferencias por Internet son otras actividades realizadas también en directo que sin *streaming* no serían posibles. Otras aplicaciones utilizan la capacidad de mezclar y sincronizar varios flujos de datos independientes como, por ejemplo, la subtitulación de vídeos. El *streaming* es realmente práctico en dispositivos con bajas capacidades como los teléfonos móviles.

La comunicación de contenidos multimedia mediante *streaming* se puede realizar de dos maneras: en directo o bajo demanda. Las marcadas diferencias entre ellas justifica el hablar de dos tipos de *streaming*: en directo y bajo demanda [78].

En el *streaming* en directo se ejecutan simultáneamente la creación de un contenido multimedia y su comunicación, todo ello sincronizadamente y en tiempo real. Esta comunicación está dirigida a uno o más usuarios que reciben la misma información en cada momento, reproduciendo lo mismo y a la vez. Estos usuarios no tienen control sobre la comunicación, no pueden detenerla o reanudarla, sólo participar o ignorarla. Esta clase de *streaming* es muy exigente siendo el retraso

máximo permitido en la comunicación muy bajo, sobre todo en aplicaciones interactivas.

El *streaming* bajo demanda se emplea para la comunicación de contenidos multimedia previamente grabados. La comunicación en este caso es individual e independiente. El usuario mantiene el control sobre ella pudiendo detenerla, reanudarla e incluso comenzar a reproducir en cualquier punto del contenido, todo ello sin tiempos de espera considerables. El nivel de exigencia del *streaming* bajo demanda es ligeramente inferior al de en directo. Esto hace que ponerlo en marcha sea más sencillo y se pueda emplear equipos con un rendimiento más modesto siempre y cuando el número de usuarios sea bajo.

El *streaming* en directo es aquel en el que se comunican contenidos multimedia, como audio o vídeo, generados en el acto, es decir, con un retraso mínimo entre la creación y la comunicación por la red pudiendo reproducirse casi al momento. Las fuentes de los contenidos son muy variadas: cámaras, webcams, micrófonos, medios de almacenamiento o instrumentos musicales compatibles con *Musical Instrument Digital Interface* (MIDI) [79] entre otras.

Cuando este tipo de *streaming* se emplea para comunicar en vivo un contenido a más de un destinatario a la vez se le suele asociar el término difusión o *broadcast*. En este caso todos ellos reciben la misma información prácticamente a la vez, con pequeñas variaciones producidas por retrasos principalmente de la red. De esta forma se puede emplear el *streaming* en directo para la difusión de contenidos multimedia de forma similar a la radio o la televisión.

Para poder efectuar *streaming* en directo no es suficiente con un servidor convencional de *streaming*. Un servidor para *streaming* en directo debe ser capaz de capturar y comprimir en tiempo real la información proveniente de la fuente del

contenido a comunicar. A este tipo de servidor se le conoce con el nombre de difusor o *broadcaster*.

El *streaming* en directo hace un uso intensivo de recursos, tanto computacionales como de comunicación. Como medida de eficiencia se suele utilizar un *broadcaster* complementado por servidores proxies ubicados en máquinas diferentes, repartiendo así la carga del proceso. Para optimizar el uso de la red se suele emplear técnicas de ahorro como la multidifusión o *multicast*, evitando al máximo la difusión individualizada o *unicast*. En cualquier caso la red nunca se sobreutiliza, no existe la posibilidad de re-comunicar partes perdidas del contenido, cosa que aunque aparentemente positiva degrada considerablemente la QoE.

El *streaming* en directo tiene múltiples aplicaciones. Las videoconferencias o la transmisión de eventos son algunas de ellas. La radio y televisión por Internet también utilizan el *streaming* en directo.

El *streaming* bajo demanda es aquel en el que se comunican contenidos multimedia ya grabados. La creación, compresión y registro de todo el contenido es anterior a su comunicación. Éste queda almacenado en un medio físico estando disponible en cualquier momento.

La relación cliente-servidor en el *streaming* bajo demanda es siempre uno a uno. La comunicación es individual e independiente, aún cuando varios clientes demandan un mismo recurso. El cliente puede ejercer el control sobre el proceso. En cualquier momento puede detener, reanudar o terminar el *streaming*.

Para poder efectuar *streaming* bajo demanda es suficiente con un servidor de *streaming* que tenga acceso directo a los recursos publicados, es decir, que éstos se ubiquen en un medio local de la máquina que ejecuta el servidor.

La eficiencia del *streaming* bajo demanda está marcada principalmente por el uso de la red. El entablar relaciones cliente-servidor uno a uno produce un gasto de recursos que se puede volver inasumible en algunos casos con un pequeño aumento del número de clientes. No es posible mitigar este problema utilizando la difusión *multicast* como en el *streaming* en directo porque ésta sólo es válida para relaciones servidor-cliente uno a muchos. Otro factor relevante relacionado con el uso de la red por el *streaming* bajo demanda es su posible sobreutilización que se da cuando se pierden partes del contenido y se vuelven a comunicar.

Las aplicaciones del *streaming* bajo demanda son numerosas y variadas. Todas ellas se basan en la posibilidad de acceso rápido a recursos multimedia que han sido almacenados en repositorios y que son accesibles por los usuarios para su disfrute mediante reproductores compatibles con este tipo de *streaming*. Ver películas o escuchar música bajo demanda por Internet, con fines lúdicos, educativos o empresariales son algunas de sus aplicaciones.

2.2 Formatos

El *streaming* de un vídeo no comprimido puede llegar a consumir un ancho de banda enorme. Sin aplicar ninguna compresión de datos previa con una resolución de alta definición (*720p HD*) (1280x720 píxeles), color verdadero (*24 bits*) y una frecuencia de muestreo de 30 fotogramas por segundo (*30 fps*) el ancho de banda requerido parte de (1280 x 720) x 24 x 30 = 663552000 bps, es decir, casi 664 Mbps lo que tanto en Internet como en redes domésticas es completamente inviable. Diferentes técnicas de compresión de vídeo [80] han conseguido disminuir en gran medida el volumen de datos a transmitir reduciendo consecuentemente el ancho de banda necesario para ello haciendo viable el *streaming* en este caso. Para un vídeo

con las características citadas y un nivel de calidad medio el ancho de banda necesario puede situarse generalmente entre los 3 Mbps y 10 Mbps dependiendo del método de compresión utilizado, lo cual queda a mucha distancia de los 664 Mbps sin compresión.

Al procesar un vídeo es necesario tener en cuenta que el incremento del nivel de compresión suele originar una degradación de la imagen que se manifiesta por medio de errores visuales. Por lo tanto es necesario un equilibrio entre un menor tamaño y una mayor calidad de imagen. Este balance entre imágenes y vídeos de alta calidad y bajo tamaño se ha ido consiguiendo progresivamente desde los años 80 y se han establecido diversas normas estandarizadas para alcanzarlo por parte de comités de expertos como el *Joint Photographic Experts Group* (JPEG) para imágenes y el *Motion Picture Experts Group* (MPEG) para vídeo.

Para que un formato de vídeo sea compatible con la técnica de *streaming* debe permitir la reproducción de cualquier fracción del contenido sin necesidad de disponer de ninguna otra anterior o posterior en el tiempo. Dos ejemplos muy usados y representativos de la técnica de *streaming* son el *Motion-JPEG* (*M-JPEG*) y el *MPEG-4*.

M-JPEG es la denominación de aquellos formatos multimedia en los que la compresión de vídeo se realiza mediante la codificación de cada uno de los fotogramas como imagenes JPEG [81].

Las características principales del vídeo M-JPEG son las siguientes:

1. *Comunicación robusta*. Cada fotograma codificado es independiente de los demás. La pérdida de alguno de ellos no condiciona la descodificación y posterior reproducción de los siguientes.

2. *Tasa de transferencia estable.* Todos los fotogramas tienen un tamaño muy parecido al estar codificados como imágenes JPEG con las mismas propiedades.

3. *Calidad de imagen estable.* Ante problemas en la comunicación de un vídeo M-JPEG la calidad de imagen no se ve alterada. Lo que sí queda afectado en estas situaciones es la frecuencia de muestreo por posibles retrasos o pérdidas de fotogramas. La calidad de imagen sólo depende del grado de compresión de las imágenes JPEG.

4. *CODEC sencillo.* La correspondencia directa de cada fotograma con una imagen JPEG facilita el proceso de conversión. Además, el formato JPEG es uno de los más utilizados en la actualidad.

M-JPEG presenta múltiples ventajas. Se consigue mayor calidad en una reproducción fotograma a fotograma. La tecnología necesaria para su manejo es simple y barata. Su proceso por medio de software fotográfico es más sencillo que con otros formatos.

M-JPEG se utiliza en aplicaciones en las que muchas veces es necesario la reproducción fotograma a fotograma lo más nítida posible, como en seguridad, vigilancia o en pruebas judiciales. Otro ejemplo se da en aquellos casos en los que el hardware es limitado y se necesita procesar vídeo en tiempo real: en cámaras de vídeo vigilancia o en consolas de vídeo juegos.

MPEG-4 es el nombre de un grupo de estándares de la *International Organization for Standarization* (ISO) / *International Electrotechnical Commission* (IEC), siguiendo recomendaciones de la *International Telecommunication Union* (ITU), y desarrollados por el MPEG para la codificación de audio y vídeo. Los

antecedentes de MPEG-4 se remontan a la aparición de la recomendación H.261 [82] que tuvo gran influencia en el posterior MPEG-1 Part 2 [83]. A continución llegaron conjuntamente H.262/MPEG-2 Part 2 [84]. Más adelante H.263 [85] sirve como punto de partida para MPEG-4 Part 2 [86], el cual da un paso más con la unión H.264/MPEG-4 Part 10 [87], la versión más avanzada de MPEG-4 que tiene como sucesora a H.265 [88].

La compresión de vídeo en MPEG-4 se basa en la división de la serie de fotogramas que lo constituye en varias subsecuencias en las que el primer fotograma de cada una, o *keyframe*, se registra en su totalidad y el resto se almacena únicamente como diferencias originadas a partir de ese *keyframe*, o lo que es lo mismo, para que el reproductor pinte cada fotograma en pantalla éste puede recibir la información completa del mismo o sólo de una parte dejando la restante sin modificar con respecto al fotograma anterior ya pintado.

Las características principales del vídeo MPEG-4 son las siguientes:

1. *Eficiencia.* MPEG-4 consigue reducir considerablemente el tamaño de un vídeo, pero a costa de que su comunicación sea más sensible a posibles errores de los que es difícil recuperarse antes del proceso de un nuevo *keyframe*. Por ello es conveniente que los *keyframes* estén lo menos espaciado posible.

2. *Tasa de transferencia variable.* La información correspondiente a cada fotograma tiene un tamaño variable dependiento de si se codifica como *keyframe* o como diferencia. Además las diferencias varían en tamaño dependiendo de si la escena tiene mucho movimiento (mayor tamaño) o poco (menor tamaño).

3. *Calidad de imagen variable.* Los problemas de comunicación pueden degradar considerablemente la imagen. La pérdida de un *keyframe* invalida todos los fotogramas asociados codificados como diferencias. Además la calidad de imagen puede degradarse si en la escena hay mucho movimiento.

4. *CODEC complejo.* La descodificación de cada fotograma no es trivial, hay que realizar cálculos a partir de los demás porque un fotograma que no es un *keyframe* está en función de sus vecinos. Un fotograma es el resultado de la suma del último *keyframe* con todas las diferencias asociadas al mismo.

Las ventajas del formato de vídeo MPEG-4 se basan en su alto nivel de compresión sin que ello conlleve una disminución considerable de la calidad de imagen. De este modo se reduce la cantidad de espacio de almacenamiento necesario, o lo que es lo mismo, con este formato caben más vídeos en un mismo medio de almacenamiento de datos. Otra ventaja es la disminución del ancho de banda requerido para las comunicaciones en Red.

MPEG-4 tiene mucha aceptación, sobre todo en Internet por ahorrar espacio en los servidores y disminuir el ancho de banda necesario para la comunicación de elementos de gran tamaño como son los vídeos.

MPEG-4 puede operar de forma similar a M-JPEG si todos los fotogramas se codifican como *keyframes*. Tal codificación es idónea si se busca estabilidad y robustez, y se dispone de un elevado ancho de banda efectivo. En cambio si los recursos de almacenamiento y de Red son limitados y se quiere dar difusión a los contenidos por Internet es más adecuado limitar el número de *keyframes*.

2.3 Componentes

La técnica de *streaming* se ajusta a los modelos de diseño cliente-servidor y P2P. En el caso básico el servidor genera el *streaming* del contenido multimedia y el cliente lo reproduce. En otros casos son necesarios otros componentes más especializados como son el servidor difusor o *broadcaster*, o el servidor representante o *proxy*.

En la Fig. 1 se ilustran los componentes mencionados con distintas relaciones entre ellos a modo de ejemplo. A continuación se analiza cada uno de ellos con más detalle.

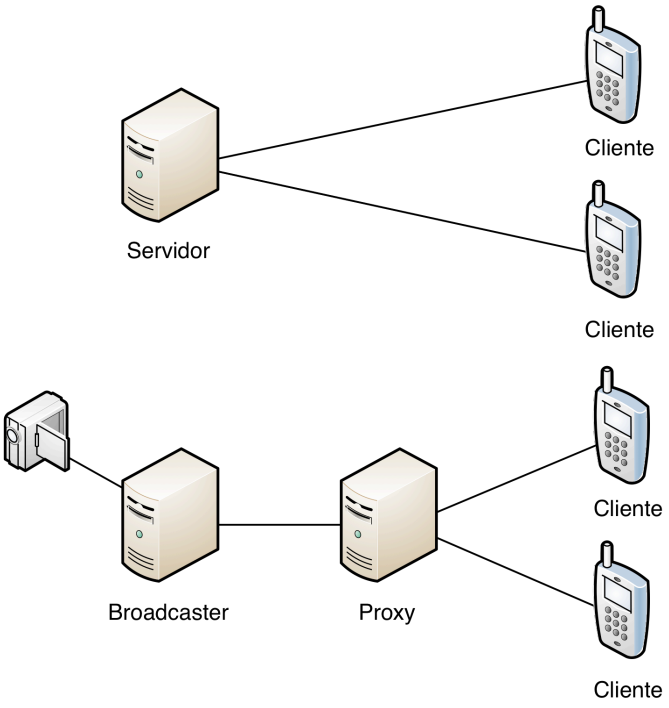

Fig. 1. Componentes del streaming

2.3.1 Cliente

Un cliente de *streaming* además de reproducir el contenido multimedia controla la sesión de *streaming*.

Un cliente de *streaming* está constituido por varios componentes:

1. *Controlador*: es el encargado de parametrizar los demás componentes para que operen correctamente. Su otro cometido es el mantenimiento de la sesión de *streaming*. Comunica las órdenes del usuario al servidor para que éste actúe según las mismas siempre dentro de sus posibilidades. Acciones de control típicas son iniciar, parar, reanudar o terminar el *streaming*.

2. *Buffer*: almacena durante un corto período de tiempo los paquetes de *streaming* recibidos. Proporciona un pequeño retraso en la disposición de los paquetes con el que se amortigua las variaciones en el ritmo de su llegada. Estos paquetes son liberados a un ritmo adecuado siguiendo un patrón FIFO. Este ritmo viene indicado explícitamente en los metadatos de los propios paquetes.

3. *Desempaquetador*: extrae el cuerpo de los paquetes de *streaming* una vez van siendo liberados del buffer. La información ubicada en el cuerpo de estos paquetes es la propia del contenido multimedia solicitado, dada en un formato de compresión.

4. *CODEC*: descomprime la información del contenido multimedia proporcionada en un formato determinado. El resultado es la información original lista para ser reproducida.

5. *Reproductor*: es el encargado de reproducir el contenido multimedia usando la pantalla, los altavoces u otro elemento requerido del dispositivo.

En la Fig. 2 se ilustra un cliente de *streaming* con los componentes citados y las relaciones que mantienen. El usuario comunica ordenes al servidor y el servidor comunica la información del contenido multimedia circula al usuario.

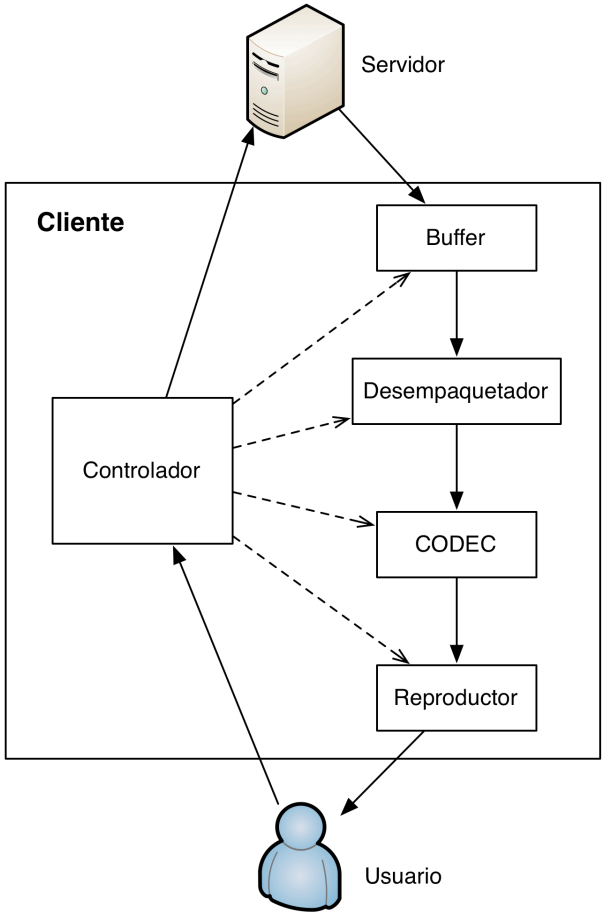

Fig. 2. Cliente de streaming

2.3.2 Servidor

Un servidor de *streaming* es el encargado de generar el *streaming* a partir de contenidos multimedia.

Los diferentes componentes de un servidor de *streaming* son:

1. *Controlador*: al igual que en el cliente, este componente es el encargado de parametrizar los demás componentes y de mantener la sesión de *streaming*. Recibe órdenes del cliente e intenta cumplirlas siempre dentro de sus posibilidades.

2. *Empaquetador*: genera los paquetes de *streaming*. En su cabecera se incluyen metadatos necesarios para la reproducción del *streaming*, y en el cuerpo cierta cantidad de información del contenido multimedia, dada en un formato de compresión. Normalmente en *video-streaming* el cuerpo de un paquete se corresponde con un fotograma.

3. *Buffer*: almacena durante un corto período de tiempo los paquetes de *streaming* generados. Su propósito es amortiguar las posibles variaciones del ritmo de empaquetado del contenido multimedia.

En la Fig. 3 se ilustra la versión más sencilla de servidor de *streaming* con los componentes citados y las relaciones que mantienen. Se puede observar cómo son comunicadas las órdenes del cliente al servidor y como la información del contenido multimedia almacenada en un medio local en un formato de compresión circula desde el servidor hasta el cliente.

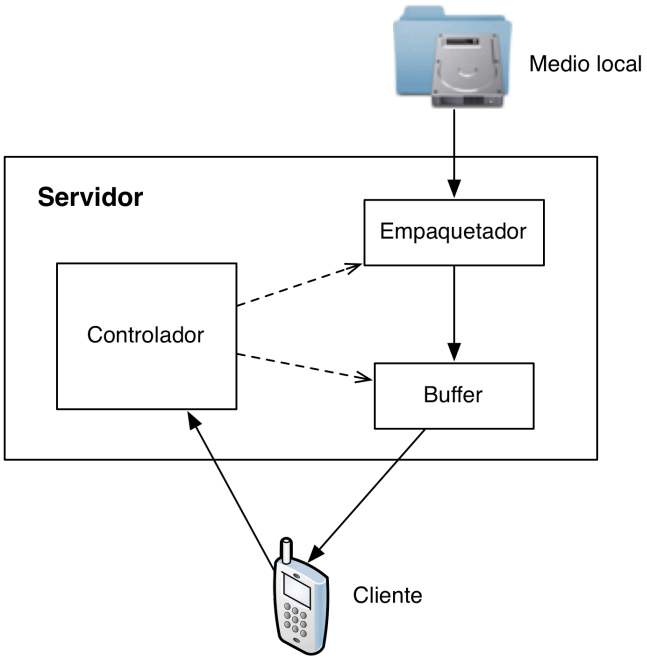

Fig. 3. Servidor de streaming

Gran parte de las ventajas que presenta el *streaming* frente a otras alternativas de comunicación de vídeo o audio a través de redes son derivadas de ciertas capacidades de un servidor de *streaming*. Estas son:

1. *Control de acceso de usuarios*. El servidor puede requerir la autenticación de los clientes para restringir el acceso a sus contenidos. También puede limitar el número de usuarios evitando la sobrecarga del servidor.

2. *Optimización del tráfico de datos*. El servidor racionaliza el tráfico de datos usando la red al mínimo. A cada cliente no se le envía el contenido a la velocidad máxima posible como en la descarga progresiva, sino a una velocidad adecuada para la reproducción. Además si el cliente detiene la reproducción el tráfico también se paraliza, cosa que tampoco ocurre en la descarga progresiva.

3. *Capacidad para mitigar efectos adversos en la comunicación.* Un servidor de *streaming* puede detectar contingencias en la Red y realizar acciones como variar la frecuencia de muestreo o la resolución del contenido adecuando así el tráfico de datos a las circunstancias existentes.

4. *Capacidad de difundir eventos en directo.* Actualmente esta tarea sólo puede llevarse a cabo en redes mediante un servidor de *streaming.*

2.3.3 Broadcaster

Un *broadcaster* es un servidor de *video-streaming* capaz de comprimir el contenido multimedia comunicado en tiempo real, a partir de una o varias fuentes externas.

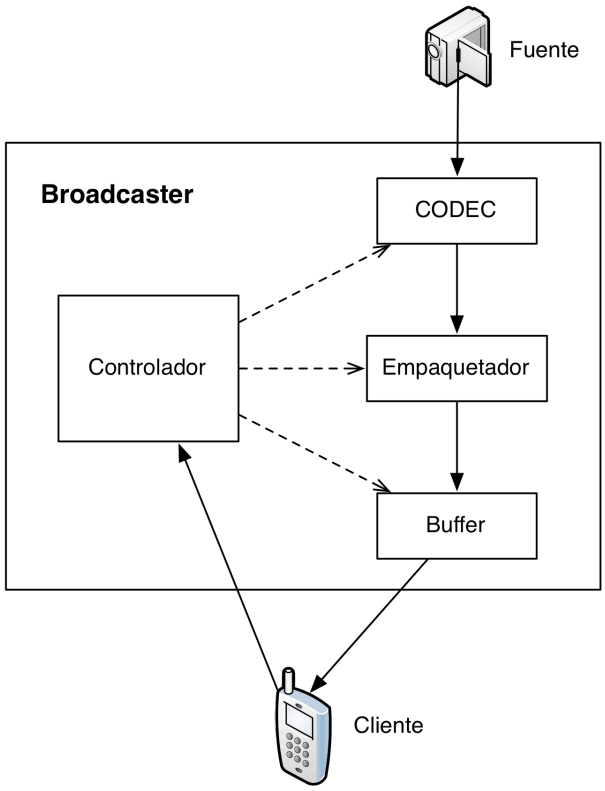

Fig. 4. Broadcaster de streaming

En la Fig. 4 se ilustra la estructura de un *broadcaster* incluyendo el CODEC de altas prestaciones necesario para sus funciones.

Para facilitar la producción en tiempo real de contenidos multimedia, con el fin de difundirlos mediante *streaming*, un *broadcaster* proporciona varias facilidades entre las que se encuentran:

1. *Codificación en varios formatos.* Para mejorar la compatibilidad con un mayor número de reproductores de *streaming* un *broadcaster* puede realizar la codificación en varios formatos. La elección de los mismos depende del público al que se pretende llegar y del tipo de dispositivos que posean.

2. *Configuración de la codificación.* Para mejorar el rendimiento de la codificación, de la difusión e incluso la de reproducción en los diferentes clientes en un *broadcaster* se pueden controlar parámetros como la resolución de imagen o la frecuencia de muestreo de vídeo u otro tipo de información multimedia.

3. *Múltiples entradas y salidas.* Un *broadcaster* puede mezclar diferentes entradas para generar el contenido multimedia resultante. Estas entradas pueden provenir de fuentes como medios físico de almacenamiento, cámaras, micros, instrumentos musicales, etc. En muchos casos la salida puede ser múltiple también, y tener diferentes formatos y configuraciones lo cual es muy útil para llegar a más público.

4. *Múltiples modos de ejecución.* Dependiendo del *broadcaster* éste puede ejecutar la difusión de sus contenidos directamente a los clientes o puede apoyarse en un servidor proxy que lo represente y realice la gestión de

usuarios y la comunicación con los clientes. Esta última modalidad se utiliza para repartir la carga de trabajo y así aumentar el rendimiento general.

2.3.4 Proxy

Un proxy de *streaming* es un tipo de servidor destinado a completar las capacidades del servidor de *streaming* al que representa. Hace de intermediario entre los clientes y el servidor representado.

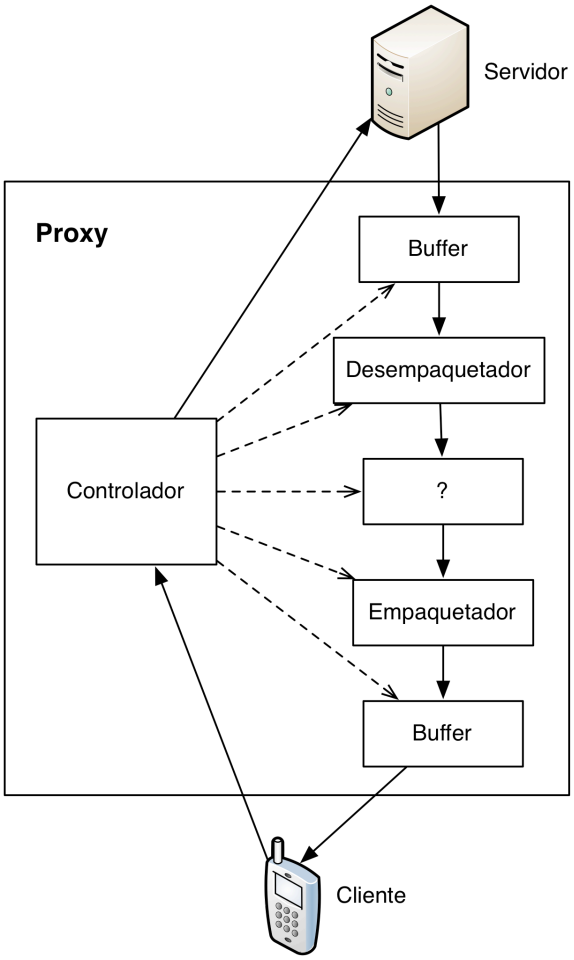

Fig. 5. Proxy de streaming

La fuente de datos multimedia de un proxy es otro servidor de *streaming*. Por ello se puede considerar que un proxy actúa a la vez como cliente ante el servidor representado y como servidor ante los clientes originales. Esto obliga a añadir componentes propios de un cliente en la estructura del servidor. Al controlador, empaquetador y buffer de salida se suman el buffer de entrada y el desempaquetador. Además puede incorporarse un componente específico para el proceso del *streaming* según el propósito concreto del proxy: añadir ciertas capacidades al servidor representado. En la Fig. 5 se ilustra la estructura de un proxy con este componente específico nombrado con un signo de interrogación.

El uso de un servidor proxy para realizar *streaming* de contenidos multimedia puede reportar varios beneficios a tener en cuenta. Entre ellos se encuentran los siguientes:

1. *Reducción del ancho de banda efectivo empleado.* Esta reducción tiene efectos muy positivos porque mitiga gran parte de los posibles problemas que se pueden dar en la comunicación de datos y que degradan la calidad del *streaming*. Las técnicas que utiliza un *proxy* para conseguir esto dependen del tipo de *streaming* que realice. Con *streaming* bajo demanda el proxy puede transmitir desde su memoria caché los contenidos requeridos si dispone de ellos, y con *streaming* en directo puede realizar una única conexión proxy-servidor aunque muchos clientes estén conectados y reproduciendo la misma difusión.

2. *Control de los contenidos.* Un *proxy* puede filtrar los contenidos disponibles para todos o para un determinado usuario, permitiéndolos o denegándolos según se estime conveniente.

3. *Seguridad*. Al unificar el acceso de distintos clientes a distintos servidores de *streaming* un proxy puede ser una herramienta para aumentar la seguridad de la red donde se encuentre. Además este acceso puede estar restringido por un control de usuarios que requiera autenticación.

2.4 Protocolos

Cuando los componentes de una aplicación necesitan comunicarse entre ellos por medio de una red de comunicación deben usar ciertos protocolos conocidos por todas las partes. Éstos permiten tanto a emisores como receptores interpretar la información compartida. Los protocolos de red son definiciones con todo detalle de la forma en la que se debe llevar a cabo estas comunicaciones.

Los protocolos determinan las estructuras de datos con las que organizar la información a compartir en el momento de su emisión y con las que interpretar esas construcciones en el momento de su recepción. En algunos protocolos también se especifica una descripción procedimental de la comunicación entre equipos usando las estructuras de datos convenidas en su caso. Con esta descripción se logra mantener conversaciones ordenadas y bidireccionales entre dispositivos dando una semántica propia a cada mensaje transmitido dentro del contexto propio de la comunicación. Así los protocolos además de marcar el formato de la información compartida indican la manera en la que realizar tal acción en cada momento de la interacción.

Para realizar *streaming* son necesarios como mínimo tres protocolos que permitan definir el perfil de la sesión, controlar el tráfico de datos multimedia y llevarlo a efecto. El perfil de la sesión especifica cómo interpretar correctamente los

datos comunicados en una sesión de *streaming*. Con esta información se parametriza adecuadamente todo el proceso de recepción y reproducción del contenido multimedia. Para el control del tráfico es necesario poder solicitar en forma de órdenes la realización de acciones específicas y poder recibir respuestas a esas peticiones por la Red. Finalmente para la comunicación del contenido multimedia en forma de pequeños paquetes de información se requiere que se permita posicionarlos secuencialmente, numerados y temporalizados.

2.4.1 Perfil de sesión

Para poder reproducir correctamente un contenido multimedia recibido mediante *streaming* es indispensable disponer con antelación de información sobre las características específicas de ese contenido y del propio flujo de datos. Esta información es lo que se conoce como perfil de sesión.

El perfil de sesión se compone de un conjunto de metadatos con los que parametrizar el proceso tanto de recepción como de reproducción del contenido multimedia. Con ello se consigue interpretar correctamente la información recibida y así reproducirla de manera satisfactoria. La difusión de un perfil de sesión puede usarse como un anuncio o invitación a la misma que permita decidir a los posibles integrantes si sumarse o no. Con la información del perfil de sesión los reproductores pueden evaluar previamente si son compatibles con el tipo de contenido multimedia de la sesión de *streaming* y si tienen a su disposición los recursos necesarios para la ejecución de todo el proceso, continuando así o abortando la incorporación sin mayores consecuencias si fuera lo adecuado. El formato estándar en Internet dedicado a la comunicación del perfil de sesión en forma de lista de parámetros es el *Session Description Protocol* (SDP) [89].

Una descripción de perfil de sesión realizada en SDP contiene la suficiente información como para que los posibles participantes puedan unirse a la sesión, o por lo menos considerar previamente si pueden hacerlo. Una descripción de sesión SDP incluye básicamente el nombre de la sesión, su propietario o creador, los intervalos de tiempo en los que la sesión está activa, los medios multimedia que comprenden la sesión, y la información necesaria para recibir esos medios (direcciones, puertos, formatos, etc.). Como datos opcionales que pueden aparecer están el propósito de la sesión, la información de contacto de la persona responsable de la sesión, el ancho de banda necesario para el tráfico de datos, la zona horaria, u otros atributos adicionales de la sesión.

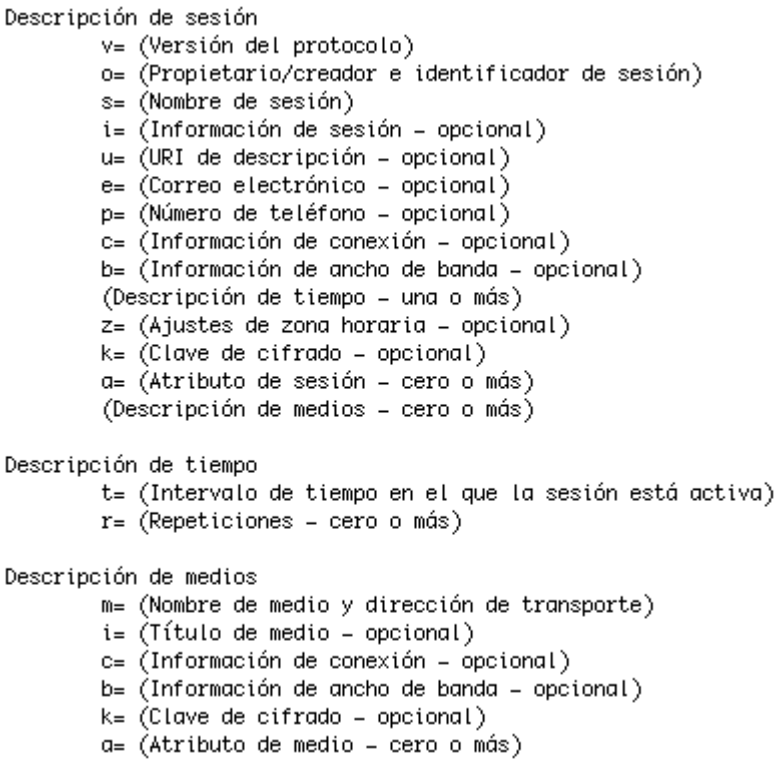

Fig. 6. Parámetros SDP

SDP se limita a especificar una estructura de datos para compartir una lista de parámetros con los que anunciar o describir sesiones multimedia (Fig. 6). En un perfil de sesión expresado en SDP cada parámetro ocupa una línea con la forma `<tipo>=<valor>` donde `<tipo>` es el carácter único que indentifica al parámetro y `<valor>` el texto que indica su valor. Un tipo especial de parámetro es el atributo que puede tomar la forma `a=<propiedad>` si expresa la existencia de la propiedad `<propiedad>` o la forma `a=<atributo>:<valor>` si proporciona un atributo `<atributo>` con valor `<valor>`. La lista de parámetros sigue un esquema con tres partes diferenciadas: descripción de sesión, descripción de tiempo y descripción de medios. La descripción de sesión es el elemento central y recoge información general sobre la misma. La descripción de tiempo indica si la sesión es ilimitada o, si no lo es, los intervalos de tiempo en los que está activa. La descripción de medios especifica detalladamente las características de cada uno de los flujos multimedia (audio, vídeo, etc.) que componen el *streaming*.

2.4.2 Control

En una sesión de *streaming* los diferentes componentes deben disponer de un mecanismo por el que interactuar entre ellos y compartir información básica. Para ello se establece un protocolo de control con diferentes órdenes como describir un contenido multimedia, comenzar o terminar la sesión, o iniciar o detener el flujo de datos. El protocolo estándar de Internet dedicado al control de sesiones de *streaming* es el *Real-Time Streaming Protocol* (RTSP) [90].

El RTSP está diseñado para mantener sesiones de *streaming* y servir de forma de comunicación de peticiones con sus correspondientes respuestas. Su uso

está orientado al control de la reproducción y la grabación de contenidos multimedia en sesiones de *streaming* tanto en directo como bajo demanda.

El formato de los mensajes RTSP es similar al de los mensajes *Hypertext Transfer Protocol* (HTTP) [91]. La primera línea indica el método requerido en la petición o el resultado en la respuesta. A continuación se ubican los campos de encabezado y finalmente el cuerpo del mensaje. RTSP se diferencia de HTTP principalmente en que introduce nuevos métodos, es dependiente del estado en el que se encuentre la sesión (HTTP no tiene estado) y tanto servidores como clientes pueden realizar peticiones (en HTTP sólo los clientes). En la primera línea de las peticiones RTSP se indica el método solicitado, el *Uniform Resource Identifier* (URI) [92] del recurso sobre el que se desea que se aplique el método y finalmente un identificador de la versión RTSP utilizada. En el caso de las peticiones la primera línea comienza con el identificador de la versión RTSP utilizada y continua con el código de estado resultante de la petición y una pequeña frase indicando la razón del mismo. Tras la primera línea aparecen los campos de encabezado. Éstos añaden atributos a los mensajes y toman la forma `<nombre>: <valor>` donde `<nombre>` es el nombre del campo de encabezado y `<valor>` su valor asignado. Según el método solicitado, o al que hace referencia una respuesta, los campos de encabezado que se incluyen son diferentes. El mensaje termina con el cuerpo, el cual es opcional. En la Fig. 7 se puede observar un esquema del formato de los mensajes RTSP.

```
Petición
        <método> <URI de petición> <versión RTSP>
        (Campo de encabezado - cero o más)
        (Cuerpo - opcional)

Respuesta
        <versión RTSP> <código de estado> <frase del motivo>
        (Campo de encabezado - cero o más)
        (Cuerpo - opcional)

Campo de encabezado
        <nombre>: <valor>
```

Fig. 7. Formato de mensajes RTSP

A continuación se describen los métodos RTSP básicos en la reproducción de contenidos multimedia vía *streaming*. Éstos son:

- *OPTIONS*: solicita una lista de los métodos disponibles y que pueden solicitarse para un recurso concreto o para un equipo, ya sea cliente o servidor, bajo las restricciones indicadas. Además de proporcionar información de cuáles son los métodos permitidos en la sesión se puede utilizar OPTIONS como "ping" con el que verificar la disponibilidad de la otra parte en la comunicación en red. El cliente es generalmente quien usa el método OPTIONS de esta manera asegurándose de que el servidor responde a sus peticiones en todo momento.

- *DESCRIBE*: proporciona la descripción del recurso indicado en la petición. El cliente especifica en su petición al servidor qué formatos de descripción es capaz de procesar. El más usado es el SDP. La descripción se ubica en el cuerpo de la respuesta al DESCRIBE y debe contener toda la información necesaria para inicializar en el cliente todos los medios descritos.

- *SETUP*: especifica el mecanismo de transporte del *streaming* del contenido multimedia solicitado. En la descripción del recurso, previamente recibida, el servidor se propone un mecanismo de transporte. Considerándolo el cliente envía un SETUP con los datos sobre el mismo que puede aceptar según sus circunstancias. Finalmente el servidor puede admitir la petición o denegarla. Si la acepta envía junto a la respuesta los datos completos del mecanismo de transporte y el identificador de sesión quedando establecida la misma, y tanto el cliente como el servidor listos para realizar el *streaming*.

- *PLAY*: una vez establecido correctamente el mecanismo de transporte con el método SETUP se puede utilizar el método PLAY para iniciar el flujo de datos. Es posible indicar el intervalo de tiempo en el que se desea que esté activo el *streaming*, en términos de tiempo absoluto o también relativo al contenido multimedia si éste es bajo demanda. Si no se especifica ningún intervalo el flujo de datos se inicia con la mayor brevedad posible y en el punto donde se detuvo antes de ejecutar el método PLAY.

- *PAUSE*: este método detiene temporalmente el flujo de datos. Como en el caso del método PLAY también es posible indicar un intervalo de tiempo en el que la acción solicitada sea efectiva, en este caso la pausa.

- *TEARDOWN*: detiene definitivamente el flujo de datos y libera los recursos asociados al mismo movilizados anteriormente por el método SETUP.

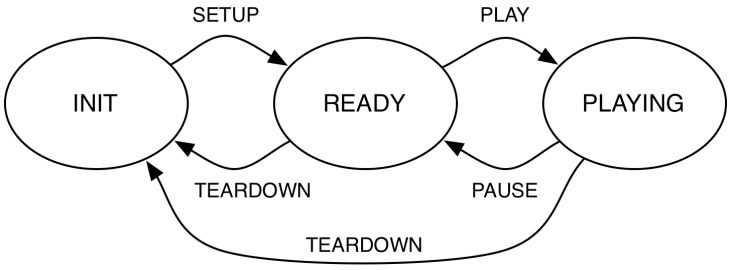

Fig. 8. Estados de una sesión RTSP

La sucesión de peticiones RTSP sigue un orden condicionado por los estados que puede tener la sesión: inicial (*INIT*), preparado (*READY*) y reproduciendo (*PLAYING*). De los métodos mencionados OPTIONS y DESCRIBE no provocan cambios de estado pero sí lo hacen SETUP, PLAY, PAUSE y TEARDOWN. En la Fig. 8 se ilustra un diagrama con los estados en los que se puede encontrar una sesión RTSP a partir de las solucitudes de estos métodos.

2.4.3 Datos

La forma del flujo de datos es el aspecto determinante en la sesión de *streaming*. Para ello existen diferentes alternativas que hay que evaluar según las circunstancias concretas de cada caso. En redes basadas en *Internet Protocol* (IP) [93] la primera elección es el protocolo de transporte a utilizar: *Transmission Control Protocol* (TCP) [94] o *User Datagram Protocol* (UDP) [95].

TCP es complejo, orientado a conexión y fiable. Garantiza que la sucesión de paquetes enviada por el emisor llegue completa y ordenada al receptor. Esto exige frecuentemente retransmisiones de paquetes ante errores o pérdidas, descarte de duplicados si los hubiere y ordenamiento de paquetes en destino. Todo ello añade retrasos y un aumento del tráfico por la red tanto de control como de paquetes

retransmitidos. TCP prima la eficacia sobre la eficiencia, es fiable pero costoso en recursos.

UDP es muy simple, sin conexión y no fiable. No garantiza la llegada de cada paquete a su destino ni mucho menos que respeten el orden de envío. No hay control de paquetes perdidos ni de errores ni de duplicidades. Esta ausencia de control repercute en un menor uso de recursos pero también en una pérdida de fiabilidad. Al no estar orientado a conexión UDP no está limitado a envíos *unicast* (uno-a-uno). Es posible realizar envíos *multicast* (uno-a-muchos) o *broadcast* (uno-a-todos) ahorrando aún más en recursos. UDP prima la eficiencia sobre la eficacia, no es fiable pero sí muy ligero y rápido.

Los protocolos de transporte no proporcionan toda la funcionalidad necesaria para mantener el flujo de datos producto del *streaming* de contenidos multimedia por lo que es necesario ampliarlos incorporando nuevos protocolos situados en un nivel superior, en el nivel de aplicación, que añadan las características necesarias a los citados protocolos de transporte. Esencialmente estos protocolos de datos para *streaming* deben proporcionar a cada paquete información sobre su orden dentro de la secuencia de paquetes y sobre su localización temporal en la reproducción. Con estas medidas es posible ordenar los paquetes, detectar posibles pérdidas, sincronizar la recepción y compensar el *jitter*.

El protocolo estándar de Internet dedicado a la transmisión de información en tiempo real que se aplica en *streaming* es el *Real-Time Transport Protocol* (RTP) [96]. Puede utilizarse sobre UDP consiguiendo un alto rendimiento en conexiones *unicast*, *multicast* o *broadcast*. RTP es un protocolo libre, abierto, completo y muy usado en aplicaciones de *streaming* lo que hace que sea muy atractivo para trabajar e investigar con él frente a otros protocolos propietarios similares que también

pueden usarse sobre UDP como el *Real Data Transport* (RDT) de RealNetworks, Inc. [97], *Microsoft Media Server* (MMS) *Protocol* de Microsoft Corporation [98] o *Real Time Media Flow Protocol* (RTMFP) de Adobe Systems Incorporated [99].

RTP incorpora a los paquetes del flujo de datos multimedia una cabecera genérica con la información mínima necesaria para llevar a cabo el *streaming* en tiempo real y para la reproducción del mismo. Esta información consta del tipo de contenido del paquete, el número de secuencia con el que ordenar los paquetes en destino y detectar posibles pérdidas si las hubiera y una marca de tiempo para sincronizar su tratamiento con la reproducción del contenido multimedia. En la Fig. 9 se puede observar detalladamente el formato de una cabecera RTP sin extensión.

El significado de cada uno de los campos es el siguiente:

- *V*: versión del protocolo.
- *P*: relleno. Indica si existen bytes de relleno al final del paquete.
- *X*: extensión. Indica si la cabecera tiene una extensión.
- *CC*: número de identificadores *CSRC*.
- *M*: marcador. Indica si el paquete tiene una especial relevancia.
- *PT*: tipo de contenido.
- *Sequence number*: número de secuencia. Este número se incrementa de uno en uno por cada paquete.

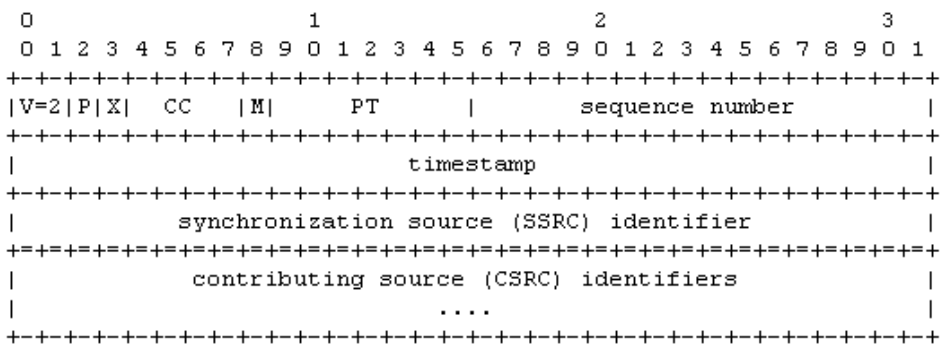

Fig. 9. Formato de la cabecera RTP

- *Timestamp*: marca de tiempo. Instante de muestreo del primer byte del contenido del paquete.
- *SSRC identifier*: identificador de fuente de sincronización. Fuente principal del flujo de datos.
- *CSRC identifiers*: identificadores de fuentes contribuyentes. Fuentes que han aportado en la generación de los datos que contiene el paquete.

La construcción del cuerpo de los paquetes RTP depende del formato del contenido multimedia original [100]. Existen especificaciones normalizadas para los formatos más usados en *streaming*. Para vídeo se pueden encontrar, entre otras, la de vídeo no comprimido [101], M-JPEG [102], H.261 [103], MPEG-1/MPEG-2 [104], H.263 [105], MPEG-4 [106], H.264 [107] o H.265 [108].

En la especificación de RTP se incluye un protocolo con el que los participantes de una sesión de *streaming* pueden distribuir información de control del tráfico de datos, el *Real Time Control Protocol* (RTCP). Este protocolo se usa principalmente para monitorizar la QoS ofrecida en cada momento. Los participantes comparten estadísticas que resumen cómo evoluciona el estado del

streaming en su posición. Otra función de RTCP es la relación de diferentes flujos de datos provenientes de un solo emisor para ser sincronizados en su reproducción. Un ejemplo de ello se da cuando audio y vídeo se transmiten en flujos separados y se reproducen simultáneamente.

2.5 Situación comercial

El análisis de la situación comercial de una tecnología indica cómo ésta es llevada a la práctica por la población en general. Para mejorarla, identificando los problemas existentes y proponiendo soluciones, tiene un especial interés conocer cómo se está usando y las tendencias que marcan en su evolución los actores más influyentes en la misma.

El *streaming* es una tecnología relativamente nueva que se ha popularizado rápidamente convirtiéndose en un elemento básico de Internet. Su desarrollo ha estado correlacionado positivamente con el aumento constante del rendimiento de las redes. Ahora mismo ese rendimiento ha alcanzado un nivel lo suficientemente alto como para que la eficiencia en las aplicaciones de *streaming* de contenidos multimedia no sea el objetivo principal en el diseño de las mismas. Esto ha dado lugar a la incorporación de técnicas prácticamente vetadas hace un tiempo pero que ahora son de aplicación plausible y provechosa al solucionar ciertos problemas. Un ejemplo de ello es el uso del protocolo de transporte TCP en *streaming*.

Utilizar TCP en *streaming* sobre redes guiadas supone el consumo constante de aproximadamente el doble de tasa de transferencia efectiva en comparación con UDP [109]. Sin embargo, se garantiza que no se pierde nada de

información en la comunicación por lo que no hay que desarrollar nuevas soluciones para este problema. Otra ventaja de elegir TCP como protocolo de transporte es la posibilidad de combinarlo con un protocolo de transacciones tan difundido como HTTP el cual está especialmente adaptado a las redes basadas en contenidos. En *streaming* HTTP resuelve el problema de la necesidad de configuraciones personalizadas de la red y los cortafuegos [110] al utilizar siempre el mismo puerto de comunicaciones. Con RTP (sobre UDP), por ejemplo, por cada canal es necesario un puerto diferente, que el servidor tenga acceso directo al cliente mediante su dirección IP y que ningún cortafuegos bloquee cada uno de los puertos de comunicación empleados. Todo ello puede solucionarse empleando HTTP a costa de un consumo mayor de recursos de red.

Cuando un cliente usando RTSP/RTP solicita el inicio del *streaming* sólo necesita comunicar la correspondiente petición y en el caso de que ésta fuera aceptada recibir y reproducir el *streaming*. Con HTTP el cliente no realiza peticiones tipo RTSP al servidor ni recibe automáticamente los paquetes. En *streaming* sobre HTTP el cliente va solicitando secuencialmente la descarga de las pequeñas partes ordenadas en las que previamente se ha dividido el contenido multimedia [111]. Esta forma de operar facilita la adaptación de la tasa de transferencia del flujo de datos a las condiciones de la red ya que el cliente en cada momento puede elegir entre diferentes versiones de una misma parte, con mayor o menor resolución y por tanto con mayor o menor tamaño, sin afectar al transcurso de la reproducción. Así el proceso de adaptación del flujo de datos mediante el control de la tasa de transferencia pasa del servidor al cliente. Esto disminuye la complejidad del servidor necesario pudiendo emplear en la mayoría de los casos uno HTTP genérico.

Pero no todo son ventajas en el *streaming* sobre HTTP. En *streaming* bajo demanda el servidor necesita más espacio de almacenamiento para guardar las distintas versiones de un mismo contenido multimedia, el cual debe estar troceado previamente y organizado en un conjunto de archivos que puede llegar a contener un número ingente de elementos. En *streaming* en vivo, el punto debil del *streaming* sobre HTTP, la exigencia en la generación del contenido multimedia en tiempo real y en diferentes versiones con distintas resoluciones simultáneamente exige una capacidad de cálculo y de manejo de datos muy alta si se desea llegar a un nivel de desempeño aceptable. Las partes generadas no pueden ser enviadas directamente sino que han de almacenarse previamente en memoria para que el servidor HTTP las pueda comunicar a los clientes una vez sean solicitadas. Todo ello va añadiendo retrasos aunque éstos pueden disminuirse si la duración de las partes generadas es menor. Esta solución hace que el número de partes aumente considerablemente y en consecuencia también el tráfico de control asociado a las solicitudes, existiendo la posibilidad de sobrepasar la capacidad efectiva de la red.

A nivel comercial el *streaming* sobre HTTP va en camino de ser la tecnología más usada con buenos resultados en redes guiadas. Existen distintos protocolos propietarios de streaming mediante HTTP destacando el *HTTP Dynamic Streaming* (HDS) de Adobe Systems Incorporated [112], el *Smooth Streaming* de Microsoft Corporation [113] y sobre todo el *HTTP Live Streaming* (HLS) de Apple Inc. [114] el cual está consiguiendo una amplia difusión. Esta tecnología también dispone de una versión estandarizada: el *Dynamic Adaptative Streaming over HTTP* (DASH) o MPEG-DASH [115].

En redes inalámbricas el *streaming* sobre HTTP no llega a satisfacer una experiencia de usuario suficientemente satisfactoria en la mayoría de los casos. Sin embargo, el uso de esta tecnología en dispositivos móviles va en aumento

progresivamente. En el sistema operativo *iOS* [116], utilizado en los móviles y tabletas de Apple Inc., desde su versión 3.0 se ha apostado claramente por el *streaming* sobre HTTP adoptándolo en exclusiva de forma nativa. El sistema operativo *Android* de Google [117] incorpora protocolos de *streaming* sobre UDP, como RTP, y sobre TCP como el *Real Time Messaging Protocol* (RTMP) de Adobe Systems Incorporated [118]. A partir de su versión 3.0 comienza a implementar *streaming* sobre HTTP, protocolo usado anteriormente sólo para descarga progresiva con HTML5. Con todo ello no existe un modelo mayoritario y que pueda solucionar todos los problemas del *streaming* con dispositivos móviles.

HTTP en *streaming* posee muchas ventajas aunque también hay que considerar su baja eficiencia que lo hace inadecuado para el *streaming* en vivo no interactivo e imposible para el interactivo. Las redes de baja capacidad quedan descartadas por la necesidad de altas tasas de transferencia efectivas. Por todo ello no se plantea HTTP como sustituto de los diferentes protocolos de *streaming* que operan sobre UDP. Éstos son una prestación constante en los servidores especializados más potentes como el *Helix Universal Media Server* de RealNetworks, Inc. [119], la compañía pionera en el *streaming* a nivel comercial, o el *Wowza Media Server* de Wowza Media Systems [120]. Sin embargo, el *streaming* sobre UDP en la actualidad sigue siendo una técnica a mejorar.

2.6 Problemática

El principal problema del *streaming* con dispositivos móviles es la elevada frecuencia de interrupciones del servicio. La causa básica es el comportamiento del medio de comunicación empleado: las redes inalámbricas. Los protocolos de *streaming* actuales no contemplan las dificultades propias de estas redes. Para

superarlas aprovechando al máximo posible la tecnología desarrollada hasta el momento es necesario introducir nuevos componentes destinados a aumentar la calidad del *streaming* con dispositivos móviles.

El carácter inestable de los canales inalámbricos hace que no se pueda garantizar que el tráfico de datos se efectúe tal y como se ha programado, o lo que es lo mismo, una QoS elevada en todo momento. Sin embargo, el *streaming* es parcialmente tolerante a cambios de QoS antes de que el usuario aprecie alguna variación en la QoE. Una disminución de QoS no implica en todos los casos una pérdida de QoE. Tampoco un aumento de QoS consigue siempre recuperar una QoE baja. Esto es debido a que la QoE no sólo se ve afectada por los efectos de la QoS.

Debido a la amplitud del concepto de QoE en *streaming* es conveniente tratarlo como una combinación de calidad de entrega o *Quality of Delivery* (*QoD*) y calidad de presentación o *Quality of Presentation* (*QoP*).

$$QoE = QoD \cup QoP$$

La QoD refleja el impacto del comportamiento de la comunicación del contenido multimedia sobre la continuidad y sincronización de su reproducción. Este comportamiento puede caracterizarse con parámetros propios de QoS medidos en el momento de entrega final de los paquetes de datos al cliente [121]. La QoP se centra en el modo en el que se realiza la reproducción del contenido multimedia. Éste viene determinado por factores como resolución, fidelidad o frecuencia de muestreo conseguida [122].

Como en cualquier servicio una elevada QoE implica confiabilidad y confort. Un servicio es confiable si da como resultado lo previsto y no falla en ello,

cumpliendo con lo que se espera de él. Un servicio es confortable si produce comodidad: sensación positiva y estable en el usuario. En *streaming* la confiabilidad está vinculada sobre todo con una QoD elevada. El confort depende de tanto de la QoD como de la QoP [123].

El ámbito de la QoD va desde la recepción de los paquetes por parte del cliente hasta la recomposición y disposición del contenido multimedia. El ámbito de la QoP va desde el proceso del contenido multimedia hasta su reproducción final. Esta distribución hace que los efectos de la QoD pueden influir en la QoP. Por ejemplo, un mecanismo como un CODEC de MPEG-4 capaz de recuperar paquetes perdidos mejora la QoD y consecuentemente atenúa la disminución de la QoP que puede originar la ausencia de partes del contenido multimedia en su reproducción. En la Fig. 10 se puede observar un esquema de los ámbitos de la QoS, QoE, QoD y QoP.

A continuación se analiza el *streaming* en redes inalámbricas, su QoD y la definición, gestión y predicción de interrupciones.

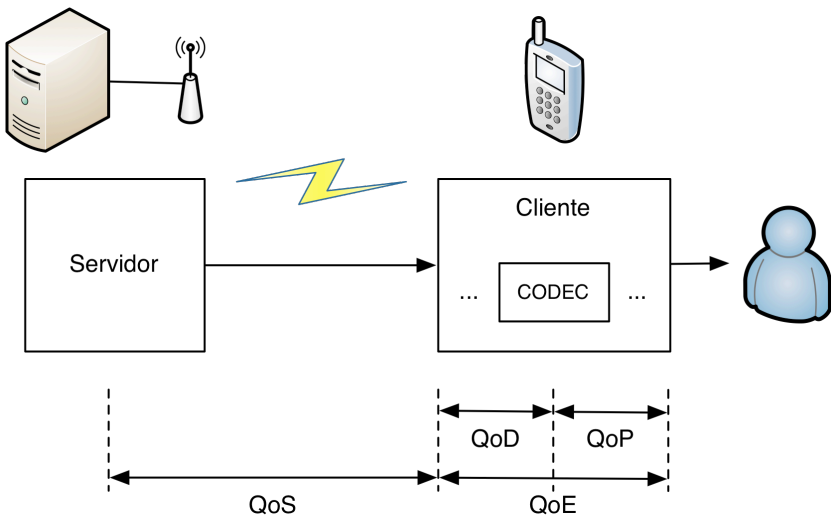

Fig. 10. Ámbito de calidades en streaming

2.6.1 Redes inalámbricas

Una aplicación de *streaming* puede ejecutarse indistintamente en redes inalámbricas o en redes guiadas. Sin embargo, aunque se empleen los mismos protocolos en cada una de ellas el rendimiento final varía drásticamente.

Los protocolos de transporte de Internet están diseñados específicamente para redes guiadas. TCP y UDP pueden emplearse en redes inalámbricas pero en ellas presentan características que dificultan e incluso invalidan el *streaming* en ciertos momentos de la comunicación. Para elegir uno de estos protocolos hay que considerar con detalle los problemas que originan en redes inalámbricas.

Para mantener el carácter fiable propio de TCP es necesario un tráfico de control constante y la retransmisión de paquetes erróneos o perdidos. En una red óptima este tráfico adicional es mínimo pero éste aumenta considerablemente en situaciones de baja calidad del enlace. En redes inalámbricas son comunes las situaciones de baja calidad del enlace por lo que el empleo de TCP puede desde llegar a ser un despilfarro hasta saturar la red. UDP no genera tráfico adicional de control y de retransmisiones pero en situaciones de baja calidad del enlace se producen pérdidas de paquetes y errores llegando a ser inasumibles con facilidad.

Para conseguir una comunicación en tiempo real con TCP es necesario mantener permanentemente congestión muy pequeña en la red y una alta calidad del enlace, condiciones que no suelen darse en una red inalámbrica. Además al garantizar TCP la entrega ordenada de paquetes si uno de ellos se retrasa los siguientes suman ese retraso y así sucesivamente. Con UDP sí se puede realizar una comunicación en tiempo real aunque a costa de posibles pérdidas de paquetes que van aumentando en número si la calidad del enlace disminuye, ya sea por congestión o por una insuficiente intensidad de la señal inalámbrica.

Para aplicaciones no interactivas si se prevé una calidad elevada y constante del enlace inalámbrico puede contemplarse el uso de TCP, si no hay que considerar UDP.

2.6.2 QoD

Una QoD elevada implica una reproducción del *streaming* fluida y sin interrupciones inesperadas y molestas. Cuantificar objetivamente la QoD en cada momento es complicado. Hay que evaluar el riesgo de interrupción de la reproducción analizando los fenómenos que pueden causarla. Éstos son numerosos y muchas veces difíciles de medir con exactitud y precisión: posición física del dispositivo, temperatura, viento, presión atmosférica, humedad, fuerza de gravedad y campos magnéticos entre otros. En la práctica considerar todas estas variables es técnicamente inviable por lo que hay que buscar otra alternativa: evaluar los efectos del estado o rendimiento de la QoS de la red sobre la QoE. Esto es posible teniendo en cuenta que a menor rendimiento de la red más riesgo de interrupción de la reproducción del *streaming* y viceversa.

Tradicionalmente se ha considerado el nivel de potencia de la señal de los dispositivos de red como elemento caracterizador del estado de la red en un punto dado. Este dato no resulta muy representativo, es bastante voluble [124] y proporciona poca información. Ante niveles similares de potencia de señal se puede encontrar situaciones muy diferentes debidas principalmente al ruido del entorno. La medición del nivel de ruido y sobre todo su relación con el nivel de potencia de la señal de los dispositivos de red, conocida como *Signal-to-Noise Ratio* (SNR), son necesarias para obtener un cierto conocimiento del estado del enlace inalámbrico. Sin embargo, a nivel individual diponer del SNR sigue siendo insuficiente para

poder sacar el máximo partido de la red llegando a la velocidad efectiva óptima en cada caso. A nivel colectivo se puede aprovechar este dato y usar métodos más avanzados como el cálculo del *Channel State Information* (*CSI*), muy útil en redes *Multiple-Input Multiple-Output* (*MIMO*) donde cohabitan múltiples receptores con múltiples emisores, consiguiendo un aprovechamiento elevado del enlace inalámbrico [125]. Aún así estas medidas son insuficientes para describir correctamente el estado del enlace inalámbrico donde existe un comportamiento habitualmente estable combinado con perturbaciones breves caóticas, generalmente producidas por interferencias o atenuaciones súbitas de la señal, que en un primer momento y en base al SNR instantáneo no pueden localizarse automáticamente. Esto requiere de un análisis más elaborado. Distinguir lo habitual de lo fortuito en redes inalámbricas es una labor muy exigente y costosa [126] aunque necesaria para mejorar este aspecto.

Las aplicaciones de los dispositivos móviles actuales, como teléfonos móviles inteligentes o tabletas, no suelen tener acceso a las mediciones de las propiedades de la red inalámbrica a nivel físico. Básicamente sólo pueden obtener en algunos casos el nivel de potencia de la señal recibida medido por el parámetro *Received Signal Strength Indication* (RSSI). Este parámetro puede ser expresado en varias unidades aunque por uniformidad y funcionalidad se viene dando usualmente por la industria en dBm, es decir, decibelios (dB) en relación a un milivatio (mW) de potencia de señal. En redes WiFi, según su especificación [8], los niveles de RSSI expresados en dBm pueden ir de -110 a 0 correspondiendose con $1 \cdot 10^{-11}$ y 1 mW. En la práctica el rango de RSSI se sitúa entre los -80 y -60 dBm. Por debajo del mismo se puede perder fácilmente el enlace aunque esto depende de la sensibilidad de la propia antena y la cantidad de ruido existente. Valores superiores muestran un enlace muy bueno y estable.

El RSSI aunque útil no resulta fiable para ningún cometido que exija estabilidad y precisión. Por ejemplo se ha intentado utilizar el RSSI como base para métodos de geolocalización a pequeña escala pero esa idea se ha tenido que reconsiderar [127]. Puede usarse el RSSI como indicación aproximada pero no como medida con la cual inferir conclusiones exactas. Por lo tanto, debido a que en los teléfonos móviles inteligentes y en las tabletas sólo se dispone del RSSI hay que buscar otra alternativa que proporcione información más exacta sobre el estado del canal: la medición directa del rendimiento del *streaming*.

El rendimiento del *streaming* está relacionado principalmente con la tasa de datos comunicados y el ancho de banda efectivo utilizado para ello. Para conocer su estado a un nivel más bajo existen otras medidas más específicas y precisas como el retraso, el jitter y la tasa de pérdida de paquetes. Estas medidas son muy utilizadas en QoS y aprovechando el conocimiento existente en este campo se puede fijar los límites que no deben traspasar en base a la percepción del usuario, o lo que es lo mismo, su relación con la QoD.

El retraso asociado a la comunicación de un paquete de datos se puede definir como el tiempo transcurrido entre el comienzo de su emisión y la finalización de su recepción. En *streaming* el impacto del retraso en la QoD varía según el tipo de aplicación. En aplicaciones interactivas éste empieza a ser apreciable a partir de 150 ms [128]. De 150 ms a 400 ms aumenta rápidamente y por encima de 400 ms es totalmente inaceptable. En aplicaciones no interactivas la tolerancia es mayor. No existen límites exactos pero se recomienta que el retraso permita que el tiempo de respuesta de la aplicación se sitúe como máximo entre 2 y 5 segundos [129].

El retraso debe ser lo más bajo y constante posible. Si éste empieza a aumentar o a variar en exceso puede ser síntoma de congestión elevada, baja SNR o bajo RSSI lo cual aumenta el riesgo de interrupción del *streaming*.

Medir el retraso con precisión es complicado. Descartando la idea de un observador único por ser impracticable en redes inalámbricas es más razonable otro método como añadir marcas de tiempo en los paquetes en el momento de su emisión (t_{emisor}) y su recepción ($t_{receptor}$). El retraso de cada paquete (D^p) es igual a la diferencia de sus marcas de tiempo.

$$D^p = t^p_{receptor} - t^p_{emisor}$$

Para que esto sea correcto no debe existir diferencia entre los valores del reloj del receptor ($r_{receptor}$) y del emisor (r_{emisor}).

$$r_{receptor} - r_{emisor} = 0$$

Si esto no es posible hay que considerar esta diferencia, que denominamos constante de sincronización (c_{sync}), en el cálculo del retraso de cada paquete.

$$c_{sync} = r_{receptor} - r_{emisor}$$
$$D^p = t^p_{receptor} - t^p_{emisor} - c_{sync}$$

Calcular la constante de sincronización es muy costoso y poco práctico cuando se requiere una precisión de centésimas de segundo e incluso milésimas de segundo. Otra alternativa para la medición del retraso es su estimación a partir del *Round-Trip delay Time* (RTT) o tiempo que tarda un paquete en llegar a su destino y volver inmediatamente a su origen. Medir el RTT es sencillo y sólo es necesario un reloj para ello (el del emisor).

El RTT se puede descomponer en retraso de ida (D^1), retraso de reenvío (δ) y retraso de vuelta (D^2).

$$RTT = D^1 + \delta + D^2$$

Suponiendo que tanto el retraso de ida como de vuelta sean prácticamente iguales y el retraso de reenvío sea despreciable es posible estimar el retraso medio en ese momento (D) a partir del RTT.

$$D^1 \approx D^2$$
$$\delta \approx 0$$
$$D \approx \frac{RTT}{2}$$

Estas condiciones ideales no siempre se dan. En algunos casos los retrasos de ida y de vuelta pueden diferir bastante por causas comunes como la saturación esporádica del canal. La inestabilidad del RTT resta fiabilidad a esta estimación del retraso. Para ello es conveniente considerar una muestra de valores RTT, filtrarla y tomar en cuenta el resultado final sólo como dato orientativo.

A partir de la estimación en base al RTT del retraso medio en un momento dado es posible estimar también la constante de sincronización (c_{sync}) suponiendo que el retraso del paquete con marcas de tiempo (D^p) sea prácticamente igual al retraso medio calculado (D).

$$D^p \approx D$$
$$c_{sync} \approx t^p_{receptor} - t^p_{emisor} - \frac{RTT}{2}$$

Hay que considerar que esta estimación es un dato inexacto y su uso provoca un error sistemático en las mediciones del retraso de cada paquete por el método de

marcas de tiempo. Este error no llega a invalidarlas y puede ser tratado a posteriori fácilmente.

La variación del retraso entre paquetes consecutivos se denomina jitter. En *streaming* los paquetes de datos se deben recibir en intervalos de tiempo regulares. Si no es así la reproducción deja de ser contínua y fluida con la consecuente disminución de QoD. Para subsanar los efectos del jitter se utiliza un procedimiento llamado compensación de retraso. Éste consiste en el almacenamiento progresivo de los paquetes recibidos en una memoria temporal o buffer para su posterior emisión transcurrido un intervalo de tiempo determinado, consiguiendo así una disposición regular. El inconveniente de la compensación de retraso es la suma de más retraso al *streaming* por lo que debe utilizarse con mesura. El jitter máximo permitido en *streaming* sin compensación de retraso depende del formato del contenido multimedia. Por ejemplo para MPEG-4 se sitúa en torno a los 150 ms [130]. Por encima de este valor es necesario emplear compensación de retraso. Como con el retraso un jitter alto puede ser sinónimo de congestión de la red o incluso de baja calidad del canal.

En la Fig. 11 se ilustra una gráfica representando como actúa la compensación de retraso. El eje de abscisas se corresponde con el tiempo y el de ordenadas con los paquetes comunicados. En ella se puede observar como por ejemplo el paquete 0 es emitido en el momento 1 y recibido en el momento 7 tras el retraso propio de la comunicación. Este paquete se almacena en el buffer y es dispuesto en el momento 10 para su reproducción.

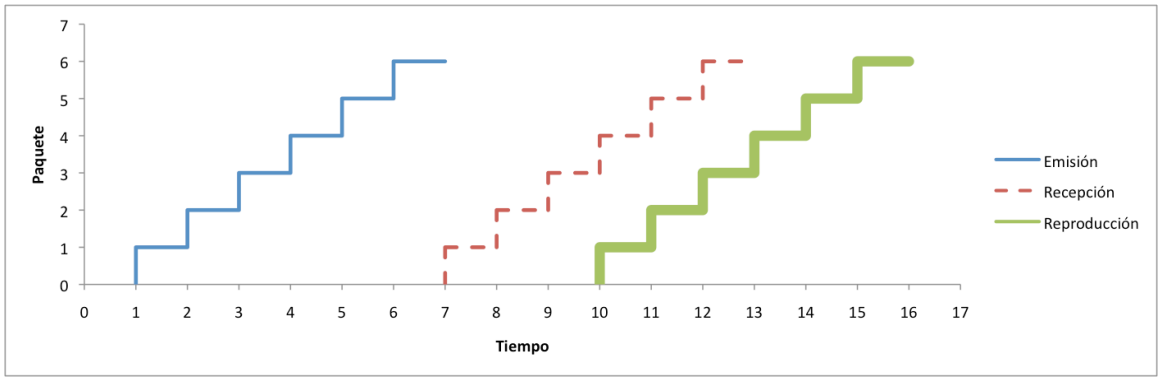

Fig. 11. Compensación de retraso

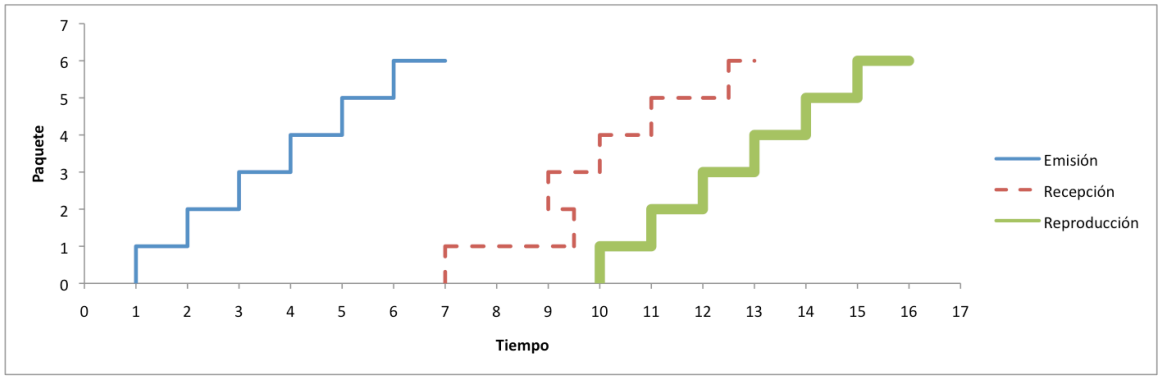

Fig. 12. Jitter

En la Fig. 12 se puede observar una gráfica similar en la que la recepción presenta jitter sin que éste logre afectar a la reproducción gracias a la compensación de retraso.

El jitter de cada paquete (J^p) es igual a la diferencia entre su retraso asociado y el del paquete contiguo anterior.

$$J^p = D^p - D^{p-1}$$

Como se puede observar la constante de sincronización (c_{sync}) queda anulada en los cálculos del jitter lo cual evita el posible error sistemático que suele introducir.

El jitter es una medida voluble y ruidosa por lo que es conveniente suavizarla. En la especificación de RTP [96] se recomienda utilizar un filtro de *Respuesta Infinita al Impulso* (IIR) de orden 1 con coeficiente $\frac{1}{16}$ para la entrada actual (J^p) y $-\frac{15}{16}$ para la salida previa (J_{filt}^{p-1}).

$$J_{filt}^{p} = \frac{1}{16}\left|J^{p}\right| + \frac{15}{16}J_{filt}^{p-1}$$

En *streaming* un paquete se considera perdido si no llega a su receptor correctamente o si lo hace demasiado tarde para su reproducción. La existencia de pérdida de paquetes puede desde generar molestias hasta invalidar totalmente la comunicación. Solventar la pérdida de paquetes requiere de retransmisiones que no siempre son posibles por las restricciones temporales impuestas por el *streaming*. Para ello es necesario detectar los paquetes ausentes, solicitar su retransmisión al emisor, que éste acepte y la realice, y finalmente aceptar esos paquetes si no se vuelven a perder, todo ello antes de que llegue el momento de su reproducción. Los formatos utilizados para *streaming* suelen estar diseñados para tolerar pequeñas tasas de pérdida de paquetes. MPEG-4 soporta adecuadamente un 3% situándose el límite en el que la pérdida de paquetes empieza a ser considerada molesta por el usuario en un 5% [131]. Por encima de este límite la reproducción habitualmente queda comprometida y la QoD decae abruptamente. La principal causa de pérdida de paquetes por retraso excesivo es la congestión de la red y por no recepción la baja SNR o bajo RSSI.

En la Fig. 13 se pueden observar los dos casos de pérdida de paquetes en una gráfica similar a la de la Fig. 11. El paquete 1 se recibe posteriormente al momento de su reproducción por lo que ésta se interrumpe. El paquete 4 no es recibido y termina produciendo los mismos efectos. En este ejemplo no se consideran retransmisiones ni un CODEC capaz de recuperar paquetes perdidos.

Ante paquetes con retraso excesivo se puede aumentar el retraso de reproducción incrementando el tamaño del buffer. En la Fig. 14 se observa como así no se pierde información aunque se interrumpa la reproducción. Este retraso queda acumulado inevitablemente a menos que el buffer sea dinámico y adaptativo [132] pudiendo reducirlo en un momento dado descartando algunos paquetes pendientes por reproducir.

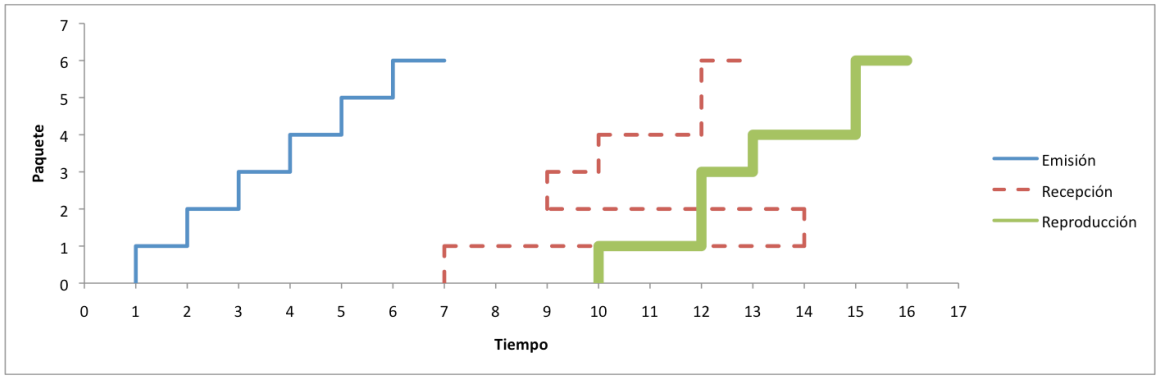

Fig. 13. Pérdida de paquetes

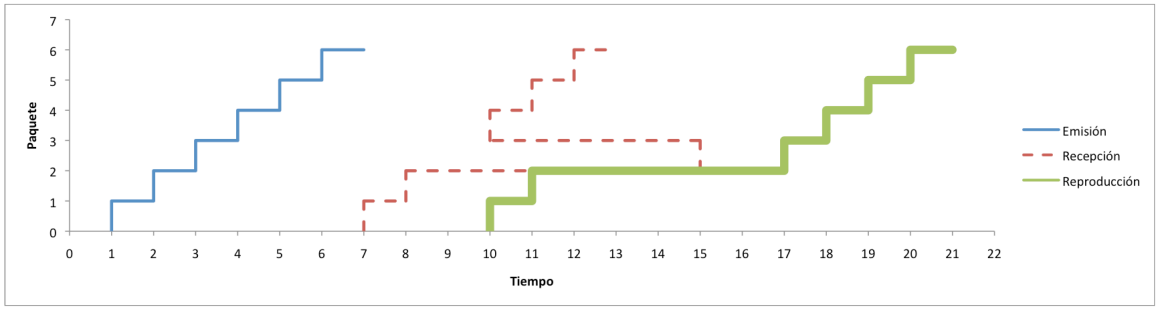

Fig. 14. Aumento del retraso de reproducción

La tasa de pérdida de paquetes (L) se corresponde con la proporción de paquetes perdidos ($P_{perdidos}$) respecto al total de paquetes enviados ($P_{enviados}$).

$$L = \frac{P_{perdidos}}{P_{enviados}} = \frac{P_{perdidos}}{P_{perdidos} + P_{recibidos}}$$

En la práctica no se suele disponer de la cantidad de paquetes enviados y perdidos. Su cálculo se suele realizar a partir del conjunto (S) constituido por los números de secuencia de los paquetes recibidos.

$$S = \left\{ p : p \in \mathbb{N} - \{0\} \right\}$$
$$P_{enviados} = Max(S)$$
$$P_{recibidos} = |S|$$
$$P_{perdidos} = P_{enviados} - P_{recibidos} = Max(S) - |S|$$

Con este método la expresión de la tasa de pérdida de paquetes puede reescribirse.

$$L = \frac{Max(S) - |S|}{Max(S)}$$

2.6.3 Interrupciones

Definimos interrupción del servicio de *streaming* como la discontinuidad fortuita de la reproducción del contenido multimedia comunicado con esta técnica. Las interrupciones se manifiestan como congelación o degradación excesiva de la reproducción y son originadas por pérdidas de paquetes prolongadas producto principalmente de desconexiones de la red.

Definimos desconexión de la red como la incapacidad propia y temporal de un dispositivo de comunicarse a través de ella por ser ésta inaccesible. Una desconexión puede clasificarse en primer lugar según la cobertura del dispositivo y a continuación según el uso de los protocolos de comunicación. Definimos tres categorías de desconexión:

1. *Física*: el dispositivo no tiene cobertura.

2. *Lógica*: existe cobertura pero se hace un uso inadecuado de los protocolos de comunicación.

3. *Virtual*: existe cobertura, los protocolos están bien empleados pero contingencias como errores, jitter elevado, retrasos o valores bajos de SNR o RSSI impiden la comunicación.

Por definición estas categorías engloban a todas las desconexiones y son mutuamente excluyentes. En la práctica esto no impide que una misma desconexión cambie de categoría a lo largo del tiempo.

Todas las desconexiones afectan a la QoS. Sin embargo, no tienen por qué afectar a la QoD y consecuentemente a la QoP. Generalmente una desconexión de la red provoca la interrupción del *streaming* siempre y cuando su duración sea elevada. Los efectos de desconexiones breves suelen ser mitigados por técnicas como la retransmisión de paquetes, CODEC capaz de recuperar los perdidos o el aumento dinámico del retraso de reproducción si es necesario.

Tras una interrupción el cliente de *streaming* puede recuperarse automáticamente o cancelar la sesión. Si es capaz de reanudar la reproducción es posible que se haya perdido información. Si es así a la interrupción pasada la denominamos corte y en caso contrario pausa. Los cortes son especialmente

agresivos con la continuidad (QoD) llegando a repercutir en la presentación (QoP) de la reproducción. Las pausas al manifestarse como congelaciones de la reproducción sólo afectan a la continuidad. Las pausas son asumibles salvo en aplicaciones interactivas o si se dan con mucha frecuencia. Los cortes casi siempre son intolerables y suponen un problema notable.

La tolerancia a los cortes depende principalmente de su duración pudiendo clasificarse en cortos (micro-cortes) si no producen una pérdida acusada de QoD y largos si lo hacen. El límite entre estas categorías tiene relación con variables referentes al estado mental del usuario y sobre todo a la secuencia visualizada. Un corte no puede ser lo suficientemente largo como para perder la percepción de coherencia de la escena. Así en una escena lenta, de baja intensidad o con poca información se tolera mejor un corte que en una rápida, intensa o con mucha cantidad de información relevante.

La medida de coherencia de la reproducción de una escena es muy subjetiva y su valor depende de la percepción de cada usuario. Al ser su cálculo objetivo demasiado complejo es necesaria otra alternativa con la que distinguir entre cortes tolerables y nocivos: considerar los límites del sistema visual humano.

Un vídeo produce sensación de movimiento a partir de 15 fps [133]. Por debajo de esa frecuencia de muestreo el usuario pierde esa sensación. Esto supone que los cortes de duración inferior a 0.067 segundos no repercuten negativamente en la QoD aunque ocurran continuamente.

La visión humana está preparada para tolerar cortes de más de 0.067 segundos si éstos son esporádicos. De forma natural se suceden continuamente por el parpadeo ocular. De media en reposo se producen 17 parpadeos por minuto [134], uno cada 3.5 segundos aproximadamente, durando cada parpadeo entre 0.1 y 0.4

segundos [135], lo que puede suponer más del 10% del tiempo total. En otras situaciones como en una conversación el número de parpadeos es incluso mayor, 26 parpadeos por minuto, sin que por ello se vea afectada considerablemente la percepción visual. Esta diferencia de 9 parpadeos por minuto puede adoptarse como número máximo de cortes tolerables. Con este criterio, los micro-cortes de *video-streaming* pueden durar como máximo 0.4 segundos y abarcar hasta el 5% del tiempo total.

Otro criterio para determinar la duración máxima de un micro-corte es el del tiempo de respuesta en interacciones persona-computador. Generalmente el usuario considera interrumpida esta interacción si en más de 1 segundo no percibe alguna señal que pruebe su existencia [136]. Esta cantidad tiene relación con el tiempo de reacción del usuario que se sitúa en unos 750 ms ante un evento esperado y 1250 ms ante un evento inesperado [137]. La sensación de interrupción aparece plenamente en el usuario si éste es capaz de reaccionar antes de que la interacción dé señales de actividad.

La combinación de estos dos criterios da como resultado que la duración de cada micro-corte esporádico puede llegar hasta los 0.4 segundos sin pérdida de QoD. A partir de 0.4 segundos la QoD se degrada hasta llegar a 1 segundo, momento en el que se vuelve intolerable. El tiempo de espera entre micro-cortes esporádicos debe ser lo suficientemente amplio como para que éstos no supongan más del 5% del tiempo total. De este modo el tiempo de espera mínimo sin degradación de QoD es igual a 19 veces la duración del último micro-corte. Por debajo de ese tiempo la QoD decae rápidamente.

Las pausas tienen como principal causa el vaciado del buffer. En ese momento el cliente de *streaming* se ve obligado a recargar el buffer (*rebuffering*) aumentando

el retraso de reproducción antes de continuar. Este proceso normalmente es apreciable por el usuario disminuyendo la QoD si bien es tolerable si no se repite con frecuencia. La asiduidad de las pausas es especialmente negativa siendo muchas veces menos molesta una más larga que varias pequeñas muy seguidas.

Las pausas pueden llegar a ser muy molestas pero siempre menos que los cortes de duración similar. En *streaming* bajo demanda los cortes son negativos pero en *streaming* en vivo fatales. En *streaming* bajo demanda puede recuperarse la información perdida volviendo a solicitar la reproducción del intervalo no recibido, pero en *streaming* en vivo la información perdida no es recuperable.

2.6.4 Gestión de interrupciones

La forma más efectiva de evitar interrupciones de *streaming* en redes inalámbricas es modificar dinámicamente la configuración física y lógica de la red y la de sus nodos. Esto requiere de un control muy complejo debido a que al mejorar las condiciones de un nodo concreto en dificultades se puede empeorar las de otros entrando en un bucle que fácilmente puede volver inestable al sistema. Además cambiar la configuración de una red aunque ésta disponga de pocos nodos tiene un alto coste. Por todo ello llevar a la práctica a gran escala esta solución es inasumible.

Otra posibilidad para gestionar interrupciones consiste en mitigar los efectos negativos éstas que producen, sobre todo los de las más graves: cortes y trenes continuos de pequeñas pausas. Esto se puede conseguir convirtiendo todas las interrupciones en pausas tolerables en las circunstancias presentes. Una solución de este tipo debe cumplir ciertos requisitos:

1. Que el cliente de *streaming* no se vea forzado a terminar abruptamente su función por una desconexión física al quedar la red inalámbrica inaccesible. Una desconexión debe pasar inadvertida tanto para el cliente como para el servidor, salvo por la inevitable pausa generada por la solución.

2. Que el cliente no pierda información en situaciones de interrupción del *streaming*. La solución debe evitar la aparición de cortes provocando una pausa cuando sea preciso. También se debe evitar de la misma manera las sucesiones rápidas de pausas cortas que aunque no provocan pérdida de información sí producen pérdida de atención y malestar en el usuario.

3. Generar el mínimo tráfico de control adicional. Un aumento del tráfico en la red puede llegar a saturarla creando un problema mayor que el que se pretende solucionar. La solución debe ser lo menos invasiva posible haciento un uso eficiente de los recursos disponibles.

4. No modificar el comportamiento del *streaming*. No incrementar el jitter ni añadir retrasos. La solución tiene que ser completamente inocua y pasar desapercibida cuando el *streaming* transcurra sin incidencias, manifestándose sólo cuando ocurran interrupciones. Su utilización no puede disminuir acusadamente el rendimiento del *streaming*.

5. Independencia y modularidad. La solución debe ser totalmente independiente formando un nuevo componente de *streaming*. Además no puede exigir cambios en el software propio de clientes y servidores, siendo válidos los existentes sin necesidad de modificaciones.

La principal dificultad para llevar a cabo esta solución es el cálculo del momento exacto en el que provocar la pausa tolerable. Esto exige determinar el límite entre

una situación normal y una crítica en la que se hace necesario actuar. Este límite es el instante en el que aparece una interrupción grave.

Una interrupción de *streaming* puede detectarse reactivamente o proactivamente. Si se hace reactivamente la interrupción es advertida posteriormente a su aparición. Es la manera más sencilla aunque conlleva riesgos: en el intervalo de tiempo transcurrido entre el instante en el que ocurre la interrupción y el instante en el que ésta se confirma se pierde información por lo que debe ser lo más exiguo posible. Si la detección se hace proactivamente la interrupción se localiza antes de que ocurra no perdiéndose información. Sin embargo, esto exige predecir interrupciones con acierto para no actuar precipitadamente o demasiado tarde.

La detección reactiva de interrupciones puede mecanizarse estableciendo protocolos específicos pero la detección proactiva exige procesos más elaborados basados en la observación de un entorno, la red inalámbrica y el *streaming*, que es, como ya se ha discutido, parcialmente observable y con un comportamiento caótico determinista.

2.6.5 Predicción de interrupciones

Para lograr predecir una interrupción hay que determinar el momento futuro en el cual se va a producir una desconexión de la red. Esto requiere la medida empírica de las variables observables relacionadas y el procesamiento continuo de esa información. Debido al inextinguible riesgo de fallo la predicción de interrupciones se corresponde con un problema de toma de decisiones en el que en cada momento hay que decidir si alertar de una posible interrupción futura o no, considerando las consecuencias que puede tener esta acción tanto si es acertada como si no.

En cada decisión la información constituida por la medida de las variables observables puede ser tratada de dos maneras. La primera de ellas se basa en la interpretación directa: según el estado actual se elige una acción determinada, la que se considere que mejoraría las condiciones existentes. La segunda se apoya en la interpretación de una predicción explícita del estado de las variables observables una vez transcurrido un intervalo de tiempo determinado. Siendo estos métodos igualmente válidos a priori se puede constatar que por norma general si en el segundo las predicciones son precisas y certeras éste produce los mejores resultados. En un entorno cambiante la interpretación directa tiene que ir adaptándose continuamente, lo que hace que nunca llegue a su máximo rendimiento. En cambio el disponer de buenas predicciones del estado futuro del entorno ante cada posible acción adoptable facilita enormemente la elección de una de ellas consiguiendo esta vez resultados óptimos.

Interpretar directamente el presente puede resultar más sencillo que realizar predicciones, las cuales, aunque se sitúen muy cerca del momento actual, son muy complicadas y mucho más en entornos caóticos como los canales de comunicación inalámbricos. Sin embargo, este enfoque tiende a ser muy rígido al asociar a cada situación posible una decisión sin considerar realmente como va a evolucionar. Aunque pueda ir adaptándose continuamente si no lo hace con la suficiente rapidez y eficacia no se consiguen buenos resultados y todo el esfuerzo queda invalidado. Tomar una decisión conociendo el futuro resulta más eficaz y no obliga a estar cambiando continuamente de criterio, aunque ello depende de la exactitud de la predicción que se utilice. Lograr buenas predicciones puede llevar a tomar las mejores decisiones posibles.

La automatización de la decisión de si alertar o no de una posible interrupción futura del *streaming* exige de una estrategia de control. Dentro de la

variedad existente en la actualidad una de las que más se ajusta a este problema es el *Model Predictive Control* (*MPC*) o control predictivo basado en modelo, y más concretamente el *Nonlinear Model Predictive Control* (*NMPC*) [138] o control predictivo basado en modelo no lineal. La actividad en una red inalámbrica se puede entender como un proceso no lineal y caótico que no puede ser linealizado por los frecuentes cambios de las condiciones iniciales de valoración de la situación. NMPC se encuentra dentro del grupo de estrategias de control óptimo que amplían el ámbito de la situación ideal perseguida representando ésta no como un conjunto de valores concretos sino como la zona de optimización del índice de costo adoptado. En este caso se busca la minimización de este índice el cual está en función de la información perdida, si la advertencia de interrupción se realiza demasiado tarde, y del tiempo malgastado, si la advertencia sucede mucho antes del momento idóneo.

NMPC opera de forma iterativa siguiendo una estrategia deslizante. En cada iteración se recogen los datos de entrada presentes que a su vez se añaden a una secuencia temporal discreta con la que se calcula el estado del sistema en un intervalo de tiempo futuro conocido como horizonte de predicción. Las señales de control son el resultado de la optimización del índice de costo aplicado a las predicciones obtenidas.

Las predicciones en NMPC se basan en un modelo matemático que es necesario elaborar. Crear un modelo no lineal válido con métodos analíticos o a partir de datos experimentales suele ser muy complicado y requerir un gran esfuerzo. Puede aprovecharse el conocimiento y las herramientas propias de campos especializados en ello, como la Sinergética [139], pero para lograrlo en el caso de este trabajo resulta técnicamente imposible. Esto es debido a que no se dispone de los valores de todas las variables independientes con capacidad de alterar el

comportamiento de la comunicación. Las variables observables son muy pocas y de esta forma no se puede conseguir un modelo matemático que reproduzca completamente el comportamiento físico del canal inalámbrico para así poder calcular y obtener una representación de la realidad futura de la comunicación con un error mínimo. Esto hace que NMPC no sea aplicable en este caso al menos en su forma original.

Como ya se ha comentado, las variables referentes al problema de las interrupciones en *streaming* que son observables en los dispositivos móviles actuales, como teléfonos móviles inteligentes o tabletas, son el RSSI, la latencia, el jitter y la tasa de pérdida de paquetes. En sí no son muy representativas salvo en sus límites admisibles ya discutidos. Otra peculiaridad es que no tienen relaciones claras entre ellas por lo que no se pueden establecer nuevas variables derivadas que aporten más información.

Un caso análogo al de las redes de comunicación se puede encontrar en las redes hidráulicas. Estas son más parecidas a las redes guiadas donde las características del flujo sí son representativas del estado de la canalización y pueden tenerse en cuenta para localizar cuellos de botella o, en el caso de las redes de comunicación, zonas de congestión. En el estudio de redes guiadas se puede utilizar esta analogía [140] ya que las pérdidas de datos, que se corresponden con las fugas de una tubería, no son lo más corriente. Siguiendo con el mismo símil, una red inalámbrica se parece más a una tubería frágil que continuamente se rompe provocando fugas frecuentes. El que ocurran estas fugas depende del entorno en que se ubique la tubería y evidentemente éste no es completamente controlable. Que el rendimiento del *streaming* sea alto es un buen síntoma pero su disminución puede tener un pronóstico dispar siendo peor la originada por las “fugas” (disminución de la calidad del enlace inalámbrico) que por la congestión de la red.

Diferenciar estas dos causas de pérdida de rendimiento sólo con el conocimiento de la intensidad de la señal de red, la latencia, el jitter y la tasa de pérdida de paquetes no da lugar a conclusiones sólidas.

En esta situación de incertidumbre un posible camino para seguir avanzando es el uso de otra estrategia de control: el control inteligente [141]. La principal característica de este tipo de control es su capacidad para enfrentarse a entornos desconocidos, adaptarse a ellos y lograr los objetivos marcados. La capacidad de autoconfigurarse y adquirir conocimiento de la experiencia [142] son la base para ello, marcando esto el nivel de inteligencia alcanzado. El mayor inconveniente del control inteligente es el tiempo empleado en la adaptación que puede ser excesivo en muchos casos o hasta infinito si ésta no consigue el nivel mínimo exigible. Otro punto a tener en cuenta en este tipo de control es que se basa generalmente en la interpretación directa, no realiza predicciones, y por tanto es difícil que consiga el rendimiento máximo perseguido.

Investigar cómo realizar predicciones aplicables en control óptimo con tan poca información puede ser una tarea productiva si se recorren caminos poco explorados hasta el momento que puedan proporcionar un cierto avance. En este trabajo se ha optado por emular un proceso biológico: la combinación intuición-pensamiento. Este proceso realiza básicamente predicciones a corto plazo, en torno a segundos, del estado de un entorno caótico como es el mundo real con un alto grado de eficacia. Además estas predicciones van mejorando con el tiempo según se vaya acumulando experiencia sin necesidad de modelos preestablecidos. Esto es ideal en el caso de la gestión de interrupciones de *streaming* en redes inalámbricas, pudiendo lograr un rendimiento mayor que el alcanzable mediante control inteligente.

2.7 Solución basada en Proxies

La solución propuesta en este trabajo es un proxy doble situado entre cliente y servidor. Las dos partes de este proxy se sitúan en lugares estratégicos para poder controlar efectivamente el *streaming* circulante por la red inalámbrica. Esta arquitectura ya se ha usado en otras ocasiones [143] y se ha probado que tiene mucha flexibilidad para incorporar diversos protocolos y algoritmos de control orientados a mitigar los efectos adversos de las interrupciones.

La situación inicial sin proxy se ilustra en la Fig. 15. En ella el cliente de *streaming* ejecutado en un teléfono móvil inteligente se conecta directamente a un servidor por medio de un enlace inalámbrico (representado simbólicamente por un rayo) con el peligro constante de interrupción. Lo que se pretende es integrar el proxy en este enlace para que haga que de cara al cliente y al servidor se comporte como uno guiado sin alterar el *streaming*.

En la la Fig. 16 se ilustra la arquitectura final mostrando las dos partes del proxy: proxy-cliente y proxy-servidor. Físicamente se encuentran separadas situándose en los extremos del enlace inalámbrico englobándolo por completo. El proxy-cliente se ubica en el teléfono móvil inteligente y el proxy-servidor en un equipo conectado al servidor de *streaming* mediante un enlace fiable, una red guiada de alta calidad preferiblemente. El enlace entre cliente y proxy-cliente también es fiable por definición al estar situados éstos en la misma máquina. En este punto la problemática asociada a la red inalámbrica desaparece para el cliente y el servidor se traslada al proxy-cliente y proxy-servidor. No es necesario realizar ninguna modificación ni del cliente ni del servidor cumpliendo con el requisito de independencia y modularidad.

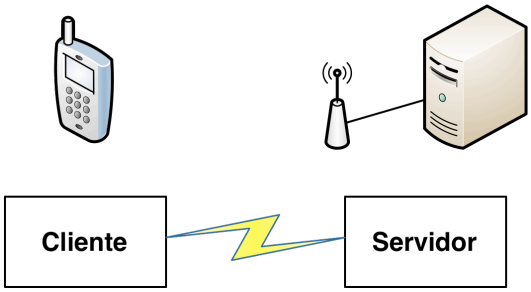

Fig. 15. Cliente y servidor de streaming

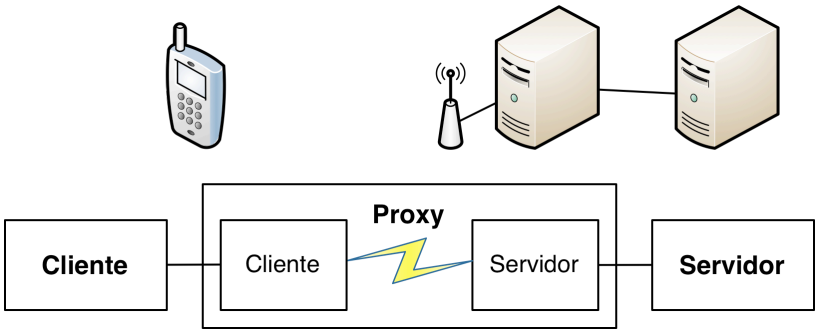

Fig. 16. Cliente, proxy y servidor de streaming

Este proxy no altera el *streaming* cumpliendo con el requisito de no modificar su comportamiento o degradarlo. Evidentemente puede añadir un pequeño retraso debido al tiempo utilizado para la gestión de los paquetes pero éste además de mantenerse constante puede minimizarse hasta hacerlo casi despreciable. En el caso de que exista una interrupción o una desconexión de la red el proxy-servidor almacena la información proveniente del servidor en una memoria, un buffer de datos, hasta que ésta pueda ser comunicada al proxy-cliente y a su vez al cliente cuando las condiciones de la red inalámbrica mejoren lo suficiente como para ello, consiguiendo así que no se pierda parte del *streaming* ante situaciones comprometidas de la red inalámbrica.

El diseño de un proxy como el descrito no sólo se enfrenta a las dificultades propias de las de uno convencional [144]. Un proxy de *streaming* tradicional opera de forma síncrona con el servidor lo cual reduce su actividad al paso de mensajes o paquetes. El proxy propuesto en este trabajo tiene que operar asíncronamente lo cual aumenta su complejidad. Esto obliga a cumplir una serie de nuevos requisitos debiendo implantar:

1. *RTSP*. Para desacoplar la sesión RTSP entre cliente y servidor ahora se tienen que crear dos nuevas: una entre cliente y proxy-cliente y otra entre proxy-servidor y servidor. Estas dos sesiones se mantienen independientes entre sí lo que obliga a tener implementados los correspondientes métodos del protocolo en las dos partes del proxy.

2. *SDP*. Para la correcta gestión del buffer es necesario conocer si el *streaming* es en directo o bajo demanda y las características de los medios implicados. Esta información se encuentra disponible en el perfil de sesión. Para interpretar correctamente las marcas de tiempo de los paquetes RTP, las cuales vienen expresadas en tiempo relativo, es necesaria la frecuencia propia del formato del medio en la que se basan. Con ella es posible calcular las marcas de tiempo absoluto con las que opera el buffer del proxy-servidor para liberar los paquetes RTP en su justo momento y al ritmo correcto.

3. *RTP*. La capacidad de analizar los paquetes RTP es básica para mantener el buffer del proxy-servidor sincronizado. Además de recoger las marcas de tiempo de los paquetes también es necesario tratar los números de secuencia con los que detectar posibles pérdidas de paquetes.

4. *RTCP*. Al igual que ocurre con la sesión RTSP en este caso también se tiene que dividir la sesión de control del *streaming* RTCP para mantener desacoplado al cliente del servidor. Los procesos de informe y mensajería propios de este protocolo deben estar implementados en las dos partes del proxy para que mantengan correctamente las dos sesiones requeridas independientes entre sí.

A parte de estos requisitos derivados del comportamiento asíncrono del proxy propuesto hay que implementar un protocolo de control interno. Las dos partes del proxy deben tener a su alcance la información sobre lo que sucede en la otra en todo momento. Esta información debe ser simplificada al máximo para no añadir tráfico adicional a la red que termine afectando a su comportamiento. Por lo tanto esta comunicación interna queda reducida al paso de eventos originados por acontecimientos esporádicos del propio proxy o del algoritmo de control del *streaming* dedicado a detectar interrupciones, desconexiones y gestionar el buffer.

El carácter flexible de la arquitectura del proxy propuesto permite aplicar diversidad de métodos de detección de interrupciones y desconexiones de la red con los que controlar el *streaming*. Sin cambios importantes se pueden usar fórmulas reactivas o proactivas, e incluso simultanearlas. Esto es ideal al emplear un mecanismo reactivo como límite de seguridad ante posibles fallos de otro proactivo. Un enfoque reactivo en este problema proporciona fiabilidad a costa de posibles pérdidas de información. Un enfoque proactivo no es completamente fiable, puesto que las predicciones pueden fallar, pero al actuar antes de que se produzca la interrupción no se pierde información. El poder compaginarlos es una ventaja positiva.

2.7.1 Protocolo reactivo

Dotar de un mecanismo reactivo al proxy propuesto como solución es imperativo para aumentar la fiabilidad. Este mecanismo prima la eficacia sobre la eficiencia por lo que siempre detecta la interrupción aunque con retraso, perdiéndose una pequeña cantidad de paquetes. En el proxy propuesto el empleo de este enfoque no es excluyente, es decir, no impide el uso de otros como el proactivo simultaneándose y complementándose.

Con la arquitectura empleada se ha probado un enfoque puramente reactivo con un protocolo inspirado en TCP y especializado únicamente en *video-streaming* [145]. Este protocolo está diseñado para no perder ningún paquete por lo que frecuentemente exige retransmisiones. Se basa en el empleo de una ventana de recepción constituida por un número determinado de paquetes. Una vez recibida una ventana si ésta es confirmada positivamente se continúa con la siguiente pero si lo es negativamente la ventana se reenvía de nuevo. En el caso de no recibir ninguna confirmación durante un corto intervalo de tiempo, más de un segundo, el proxy considera que se encuentra ante una interrupción y comienza a guardar el *streaming* en el buffer. Este protocolo es más apto que TCP por considerar las interrupciones y desconexiones pero continúa con defectos típicos como la no diferenciación entre congestión y disminución de la calidad del enlace, o las retransmisiones que bloquean el *streaming* y obligan al cliente a detenerse y volver a cargar su buffer de reproducción para continuar [146]. Con este método se exige gran cantidad de tráfico de control y se añaden los retrasos necesarios para realizar las confirmaciones.

El mecanismo reactivo propuesto para el proxy de este trabajo es mucho más simple: está especializado sólo en la detección de desconexiones de la red. Para ello

el proxy-cliente tiene la obligación de hacer *ping* al proxy-servidor si transcurre más de un cierto intervalo de tiempo sin que el primero envíe algún tipo de mensaje al segundo. Así el proxy-servidor puede cerciorarse de que el proxy-cliente está activo y conectado. Este intervalo es variable según las condiciones del enlace ahorrando así en tráfico de control.

2.7.2 Monitorización proactiva

El protocolo reactivo utilizado en el proxy planteado es muy básico y sólo detecta desconexiones de la red. El peso del control del *streaming* y del enlace inalámbrico reside en un mecanismo de monitorización. El protocolo reactivo tiene como función servir de elemento de seguridad en condiciones críticas cuando la monitorización queda invalidada.

Esta monitorización, a partir de las variables observables ya estudiadas (RSSI, retraso, jitter y tasa de pérdida de paquetes), tiene como objetivo final el tomar automáticamente y en todo momento la decisión de iniciar o no el almacenamiento del *streaming* en el buffer ubicado en el proxy-servidor. Esta decisión depende de si una interrupción es inminente o no. Se actúa con anterioridad intentando que el intervalo de tiempo entre el comienzo del almacenamiento del *streaming* y la propia interrupción sea mínimo.

Como ya se ha discutido, la disyuntiva en el diseño de un mecanismo automático para esta toma de decisiones se sitúa entre el empleo de control inteligente o un nuevo tipo de control predictivo similar al MPC pero de inspiración biológica que evite la necesidad de disponer de un modelo matemático explícito de las redes inalámbricas. La alternativa más sencilla es el control inteligente con el que simplemente se interpreta el momento presente en base a un

criterio y se obtiene como resultado la decisión buscada. Este criterio tiene que adaptarse continuamente y se corre el riesgo de que no lo consiga a la velocidad requerida, es decir, no converja y en consecuencia el sistema se vuelva inestable, perdiendo por completo la QoE. Para evitar esto se puede renunciar a un rendimiento óptimo y seguir políticas conservadoras con criterios más restrictivos que garanticen cierta estabilidad. Esto en un entorno caótico nunca es recomendable y en la práctica garantiza una baja aceptación por parte del usuario al sentirse restringido y limitado. Esta situación justifica dedicar un esfuerzo en la segunda alternativa de control planteada.

Para llevar a cabo la monitorización proactiva del proxy se propone implementarla bajo el paradigma de los agentes software. Éste es especialmente apto para este tipo de problemas distribuidos al proporcionar capacidades avanzadas de autonomía y comunicación que facilitan tareas como la paralelización de procesos ejecutados en varios dispositivos remotos, la distribución de la carga computacional de esos procesos o el escalado de sistemas entre otras. Utilizando la misma arquitectura de proxy-cliente y proxy-servidor se ha desarrollado un prototipo para recuperar automáticamente la sesión de *streaming* con agentes software [76] pero sin mecanismo de monitorización y encapsulando todo el tráfico de control y de datos como mensajes entre agentes. En la Fig. 17 se ilustra un esquema de este sistema.

Esta solución aprovecha los mecanismos de reconexión y los buzones de mensajes de los agentes, muy útiles para el problema, pero la forma de canalizar el tráfico empleada es poco eficiente debido a que los agentes se comunican utilizando TCP. Evidentemente según el planteamiento inicial el tráfico de datos tiene que ir por UDP. Para seguir utilizando agentes software pero con tráfico de datos UDP se puede modificar el sistema quedando como se muestra en la Fig. 18. En este caso

los agentes software gestionan directamente el tráfico de control y supervisan una nueva versión de los componentes proxy-cliente y proxy-servidor dedicados exclusivamente al tráfico de datos. Para la monitorización proactiva el agente proxy-cliente adopta el rol de observador mientras el agente proxy-servidor asume la toma de decisiones en base a la información proporcionada por el agente proxy-cliente, acaparando la mayor parte de la carga computacional de la monitorización y liberando al dispositivo móvil de parte de ella.

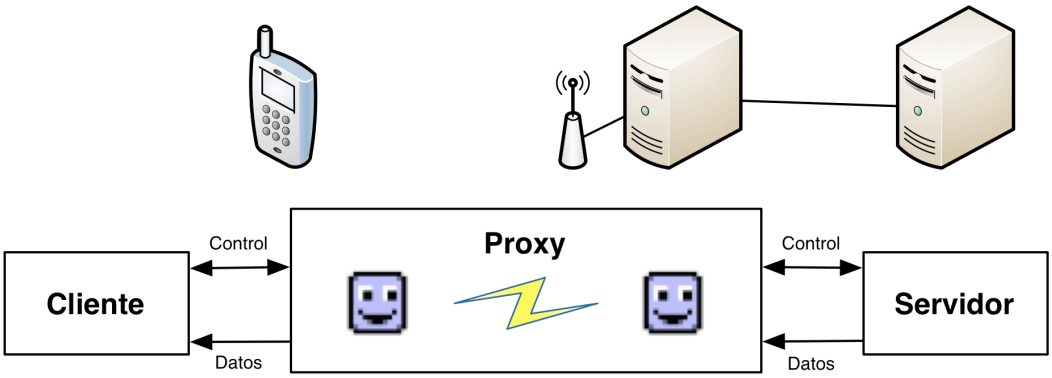

Fig. 17. Proxy con streaming entre agentes

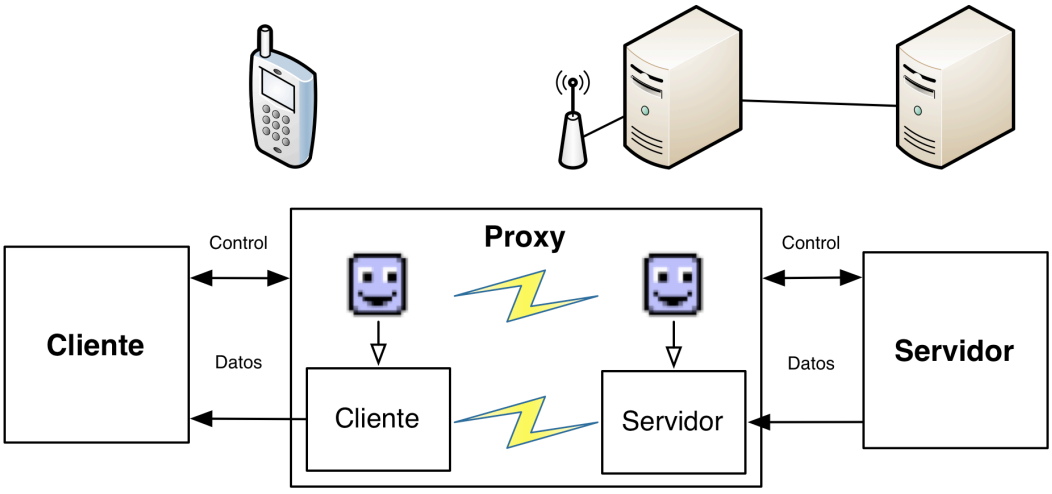

Fig. 18. Proxy con streaming externo a los agentes

Para que la monitorización no disminuya el rendimiento de la red ésta no debe suponer un aumento del tráfico total. Con el proxy el tráfico RTCP se mantiene entre cliente y proxy-cliente y entre servidor y proxy-servidor pero no entre proxy-cliente y proxy-servidor quedando esa cuota no consumida disponible para la monitorización realizada por los agentes. Mientras esta cuota no se rebase no habrá un consumo mayor de recursos de red.

Existe la posibilidad de que el proxy planteado aumente la cantidad de tráfico total en algunos momentos pero por lo general esto no ocurre. Con RTCP tanto la parte cliente como la servidor emiten informes que se van repitiendo por cada medio con un período no inferior a 5 segundos [147]. La monitorización efectuada por los agentes exige una frecuencia de muestreo mayor por el carácter inestable de las redes inalámbricas que pueden cambiar de estado rápidamente. En cuanto a tráfico este punto queda compensado en parte al sólo ser necesaria la comunicación de informes en un sentido, de agente proxy-cliente a agente proxy-servidor, y a que estos informes además de reducidos son globales y no propios de cada medio. El período de esta comunicación es variable y depende del algoritmo de control utilizado buscando siempre que sea lo mayor posible.

En la Fig. 19 se ilustra un esquema de las comunicaciones mantenidas por el proxy con los protocolos empleados. Para la comunicación de datos entre el proxy-servidor y el proxy-cliente se emplea un nuevo protocolo interno: *Procolo de Datos del Proxy* (*PDP*). Sus funciones son multiplexar los diferentes medios del *streaming* y facilitar la monitorización. Este protocolo añade en cada paquete un número de puerto destino, un número de secuencia global y una marca de tiempo absoluto. Los mensajes de control, monitorización y notificación de eventos son encapsulados y comunicados como mensajes entre agentes.

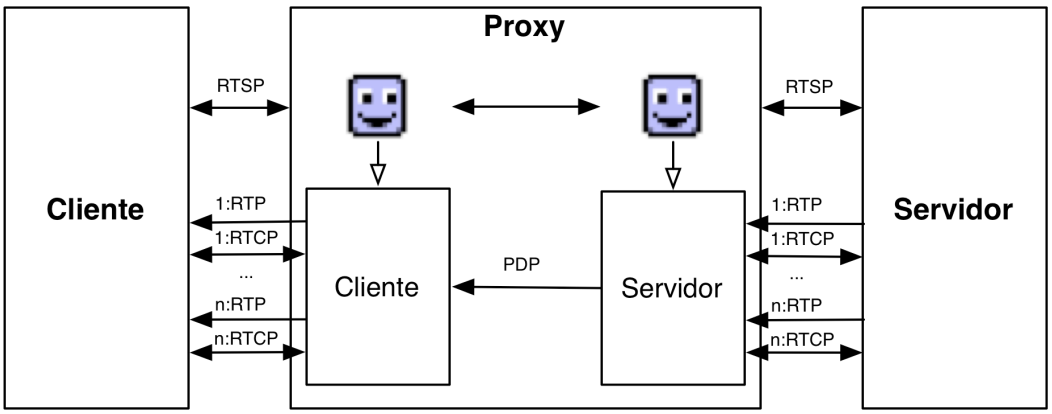

Fig. 19. Detalle de comunicaciones del Proxy

Para realizar esta monitorización proactiva y predecir interrupciones los agentes empleados en el proxy soportan el modelo de la combinación intuición-pensamiento propuesto en este trabajo. Estos agentes de nuevo diseño son los agentes software emocionales.

Capítulo 3

Agentes software emocionales

Resumen: este capítulo se centra en los avances realizados para dotar a agentes software de cualidades de inspiración biológica como la intuición, el pensamiento y las emociones. Éstas les permiten predecir el comportamiento de sistemas caóticos bajo un horizonte reducido pero suficiente para actuar con antelación ante interrupciones en la reproducción del video-streaming.

3.1 Introducción

El campo abarcado por los agentes inteligentes es realmente heterogéneo, fruto de la combinación de un amplio grupo de disciplinas entre las que destaca su gran predecesora: la inteligencia artificial distribuida [148]. En ella un agente es una entidad diseñada para tomar decisiones continuamente a partir de la percepción de su entorno. A su vez un agente puede asociarse a otros formando una sociedad con la que enfrentarse a problemas mucho más complicados e imposibles de solucionar de forma individual. En este caso esta sociedad de agentes se denomina sistema multi-agente. Dentro de un sistema de este tipo cada agente puede tener capacidades diferentes, información distinta, objetivos comunes y muchas veces hasta intereses opuestos [149]. Esta variabilidad hace que los sistemas multi-agente sean muy flexibles y aptos para la resolución de problemas distribuidos complejos y también para la elaboración de modelos y simulaciones sociales.

Los agentes software inteligentes, o simplemente agentes software, son una particularización del concepto general de agente inteligente. Un agente software es un programa informático con las características de un agente inteligente. La aparición de los agentes software data de la década de 1970 [150] donde fueron definidos como "objetos autónomos, interactivos y de ejecución concurrente, poseedores de estado interno y de capacidad de comunicación" [151]. En la década de 1990 su difusión aumenta considerablemente con los inicios de un nuevo paradigma de programación: la programación orientada a agentes [152]. En este paradigma se amplia el concepto de agente software proporcionandoles cualidades mentales como creencias, desiciones, capacidades y obligaciones. A partir de ahí la investigación en este campo aumenta, abarca más áreas de conocimiento y sigue en continuo avance en la actualidad.

Los agentes software tienen multitud de aplicaciones [153], tanto académicas como comerciales, de las que se ha ido adquiriendo experiencia [154] con el tiempo. Aunque existen problemas, sobre todo de seguridad [155] y confiabilidad [156], esta es una tecnología en expansión y con futuro.

3.2 Definición

Actualmente no existe una definición formal consensuada de agente software. Muchos autores tienen la suya propia [157] pero todavía no se ha llegado a proponer una completamente satisfactoria. Generalmente se considera agente software a una entidad contenida en un programa informático que dependiendo de su percepción y su conocimiento almacenado emprende aquellas acciones que estima más adecuadas para maximizar su rendimiento dentro de su entorno.

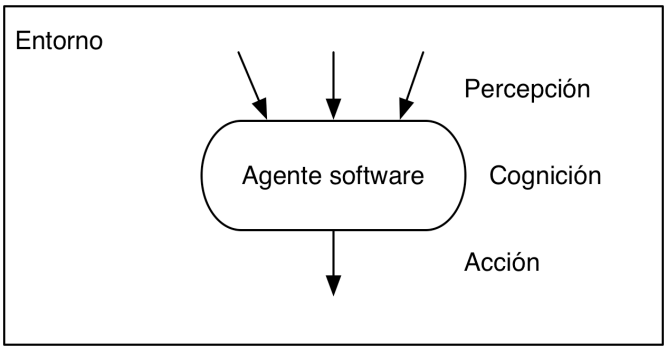

Fig. 20. Agente software en su entorno

En la Fig. 20 se muestra de forma esquemática un agente software situado en su entorno. Éste es percibido por el agente y su cognición marca la acción a tomar para modificarlo dentro de sus posibilidades buscando siempre lo que considere como éxito. Esencialmente un agente software se basa en la ejecución continua del bucle percepción-cognición-acción.

Los agentes software tienen una serie de características básicas comunmente aceptadas, las cuales los diferencian de otros elementos informáticos. Estas son la autonomía, sociabilidad, reactividad y proactividad [158].

Un agente software es autónomo al tener capacidad de operar sin dependencias para conseguir sus objetivos a partir de la información de que disponga, sin necesidad de entidades externas como personas u otras que lo controlen directamente.

Un agente software es sociable. Aunque puede actuar autónomamente también puede comunicarse e interactuar con otros agentes o personas por medio de un lenguaje inteligible por todas las partes. Su entorno ideal es un sistema multi-agente en el que no se ejecuta de forma aislada sino en conjunto con otros agentes que colaboran entre sí para realizar un trabajo.

Un agente software es reactivo. A partir de los estímulos externos que percibe de su entorno modifica su comportamiento para así lograr sus objetivos, es decir, en base a su percepción del estado del entorno donde se ubique puede reaccionar determinando en cada momento las acciones correspondientes a tomar.

Un agente software es proactivo y no sólo responde a su entorno. Puede tomar la iniciativa y realizar acciones como resultado único de sus propias decisiones sin intervención de estímulos externos. Por lo tanto un agente software es capaz de modificar su comportamiento para conseguir sus metas en base a sus percepciones (reactividad), a su conocimiento (proactividad), o a una combinación de ambos.

Para que un agente software sea autónomo, sociable, reactivo y proactivo debe poseer un cierto grado de inteligencia y adaptabilidad. Debe tomar decisiones racionales y aprender de su experiencia constantemente. Debe emprender las acciones que estime más convenientes para lograr sus objetivos, basándose en la percepción de su entorno y el conocimiento adquirido con anterioridad. Gracias al aprendizaje consigue adaptarse a la situación en la que se encuentre y mejorar las posibilidades de alcanzar sus metas. La combinación de cualidades mentales para implementar esta inteligencia, como creencias, intenciones y obligaciones, es común en los agentes software [152], llegando hasta incorporar emociones en algún caso [159].

Otra característica indispensable de un agente software es su persistencia: se ejecuta contínuamente. Sin embargo, éste no tiene por qué hacerlo siempre en el mismo lugar. A la cualidad que le permite a un agente software moverse físicamente por distintos nodos de una red se la denomina movilidad, y sólo la presentan algunos, es opcional. La movilidad de los agentes software reduce el

empleo de la red al producirse la comunicación sólo en el interior del equipo anfitrión. Una vez realizada la tarea del agente móvil éste puede regresar a su lugar de origen si es necesario. Otra ventaja de la movilidad es que los agentes software al desplazarse amplian su campo de actuación llegando a puntos inaccesibles en un primer momento en los que pueden realizar acciones o mantener conversaciones con agentes con los que les era imposible.

Los agentes software son la base conceptual de la tendencia actual del software de última generación, cada vez más complejo y autónomo, que exige la mínima interacción posible con sus usuarios pero a su vez resultados óptimos en entornos complejos [160].

Los agentes software pueden clasificarse de diversas maneras. Una de ellas es en base al grado de desarrollo de sus propiedades básicas. De este modo según su comportamiento se puede considerar a un agente eminentemente reactivo o proactivo, o según su movilidad estático o móvil.

Dependiendo del grado de inteligencia y de la forma de actuar de los agentes software se pueden distinguir las siguientes categorías [161] ordenadas de menor a mayor complejidad:

Agente reactivo simple. Este tipo de agente utiliza reglas estímulo-respuesta con las que la conexión entre percepción y acción es directa. El agente simplemente busca la regla cuya condición se ajusta a lo que percibe en cada momento y ejecuta la acción correspondiente. Este tipo de agente sólo es válido en un entorno totalmente observable donde se puede conocer toda la información necesaria sobre su estado con la que aplicar correctamente y sin ambigüedad las reglas. En la Fig. 21 se puede observar la representación gráfica de la estructura de un agente reactivo simple y cómo lo percibido es emparejado con las reglas disponibles para

dar como resultado la acción a realizar. El proceso de emparejamiento es simbolizado con una 'X'.

Agente reactivo basado en modelo. Este agente supera la limitación del agente reactivo simple logrando interacturar con efectividad en entornos parcialmente observables. Para ello usa un modelo matemático del entorno con el que consigue complementar la información parcial que es capaz de percibir. Así puede continuar aplicando la reglas pertinentes sin ambigüedad. En la Fig. 22 se ilustra la representación gráfica de la estructura de un agente reactivo basado en modelo y cómo lo percibido es completado por medio de un modelo para ser emparejado con las reglas disponibles que dan como resultado la acción a realizar. El proceso de emparejamiento es simbolizado con una 'X'.

Agente basado en objetivos. Este tipo de agente amplía las capacidades del agente reactivo basado en modelo orientando su comportamiento no al cumplimiento de un conjunto de reglas sino al logro de unos objetivos. Estos objetivos vienen representados por sucesiones de estados del entorno deseables que tienen que conseguirse sucesivamente describiendo la guía del comportamiento del agente. Éste busca la mínima desviación de esta guía eligiendo la acción que estime más adecuada en cada momento de entre todas las que pueda realizar. Para llevar a cabo esta elección emplea técnicas de toma de decisiones. En la Fig. 23 se ilustra la estructura de un agente basado en objetivos el cual a partir del estado actual de su entorno elige la acción a realizar, de entre todas las posibles, cuyas consecuencias le acerquen más a sus objetivos perseguidos. El citado proceso de decisión es simbolizado con un '?'.

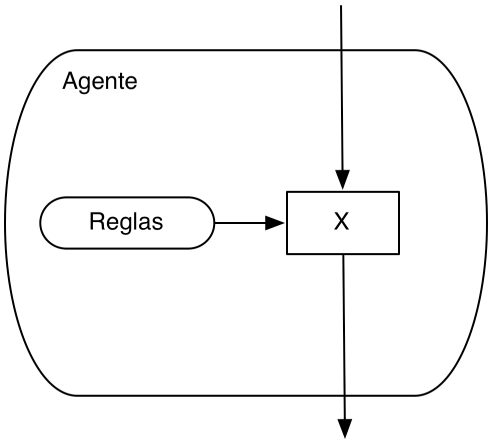

Fig. 21. Agente reactivo simple

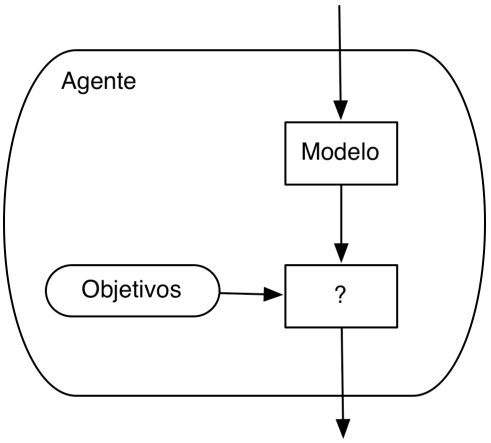

Fig. 23. Agente basado en objetivos

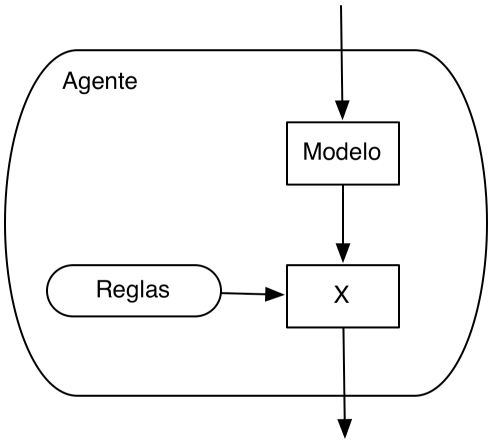

Fig. 22. Agente reactivo basado en modelo

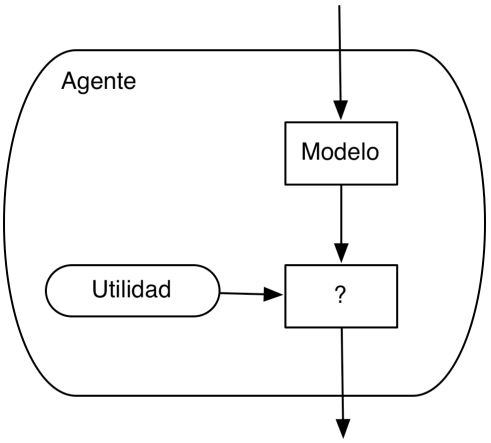

Fig. 24. Agente basado en utilidades

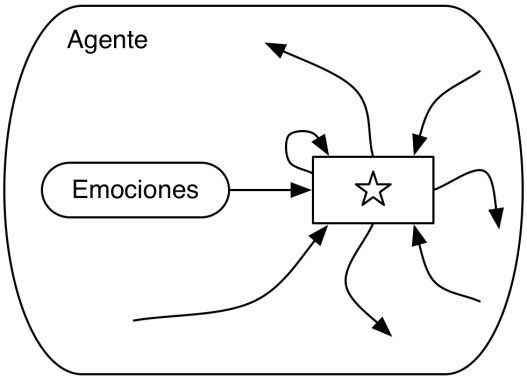

Fig. 25. Agente emocional

Agente basado en utilidades. Una versión más compleja y avanzada del agente basado en objetivos es el agente basado en utilidades. Para el agente basado en objetivos sólo existen estados del entorno objetivo y no objetivo. Esta visión extremista hace que siempre busque el camino más corto para alcanzar sus objetivos. Esta estrategia muchas veces no es la más eficaz. Un agente basado en utilidades planifica como llegar a sus objetivos. Para ello se ayuda con la llamada función de utilidad que asocia a cada estado del entorno una medida de cuán deseable resulta para el agente. De esta forma logra comportamientos más acertados y puede resolver conflictos producto de objetivos simultáneos y contradictorios. Un agente basado en utilidades elige la acción a realizar que tenga como consecuencia esperada el estado del entorno con mayor valor de utilidad. En la Fig. 24 se ilustra la estructura de un agente basado en utilidades en el que una vez conocido el estado del entorno actual elige la acción a realizar, de entre todas las posibles, cuyas consecuencias tenga una mayor utilidad. Este proceso es simbolizado con un '?'.

Un agente basado en utilidades situado en un entorno caótico no suele dar resultados satisfactorios. Esto se debe a que un modelo adecuado para este tipo de entorno es difícil de elaborar. Esta limitación no tiene una solución sencilla. Una posibilidad es aplicar técnicas de inspiración biológica que doten al agente de las cualidades que hacen que los seres vivos y especialmente los animales puedan interactuar en entornos caóticos con eficacia. La más destacada para ello son las emociones.

El término agente emocional resulta ambiguo y generalmente se aplica a todo agente que presente o utilice el concepto de emoción [162]. Por ello proponemos un agente emocional, ilustrado en la Fig. 25, que deriva del agente basado en utilidades y que posee ciertas características adicionales de inspiración biológica:

- *Cuerpo*. El cuerpo de un animal es su parte física y está constituida por sus sistemas orgánicos. Es el medio por el cual percibe su entorno y actúa modificándolo. La percepción de un animal se limita a su cuerpo y para conocer su entorno se basa en cómo éste afecta a su cuerpo. Para un agente emocional un cuerpo virtual supone una mayor cantidad de información disponible sobre su entorno aunque ésta sea indirecta y subjetiva. Esto palía en parte la pérdida de información que provoca la ausencia de modelo en su arquitectura.

- *Intuición-pensamiento*. Un animal combina su percepción con su experiencia para predecir qué va a pasar inmediatamente. Con esta predicción toma las mejores decisiones posibles en busca de emociones positivas. Este proceso se corresponde con la combinación intuición-pensamiento. Un agente emocional sustituye el concepto de utilidad por el de emoción e incluye intuición-pensamiento virtual, simbolizado por una estrella en la figura, que le permite además de tomar decisiones utilizar la experiencia acumulada como modelo del entorno.

3.3 Sistemas multiagentes

Los agentes software son realmente útiles cuando forman sociedades, o lo que es lo mismo, sistemas multiagentes. La interacción entre ellos les brinda la capacidad de resolver problemas conjuntamente que de forma individual no pueden por limitaciones de capacidad, espaciales, perceptuales o de otro tipo. Para ello emplean un lenguaje de comunicación entre agentes o *Agent Communication Language* (ACL) con el que consiguen comunicar información y conocimiento.

En un sistema multiagente múltiples agentes software se ejecutan concurrentemente para resolver problemas complejos de manera distribuida comunicándose y dialogando unos con otros generalmente mediante el paso de mensajes. Gracias a esta comunicación pueden compartir información, coordinarse, colaborar, descomponer el trabajo y negociar para resolver conflictos. Con todo ello y teniendo en cuenta que cada agente posee una capacidad insuficiente y una información incompleta para solucionar el problema global éstos consiguen controlar el sistema de forma descentralizada decidiendo individual y autónomamente las tareas a realizar para conseguir sus objetivos.

Un sistema multiagente queda definido por los agentes que lo forman, el entorno donde éstos se ubican, y sobre todo por las relaciones que mantienen entre ellos. Los sistemas multiagente son sistemas descentralizados donde cada agente tiene sólo parte de la información disponible. No existe una autoridad global que proporcione esta información a todos los agentes y normalmente un agente no puede interaccionar con todos sus congéneres, sólo con sus vecinos, por lo que tampoco tiene a su disposición toda esta información. Esta situación de observabilidad y control parcial obliga a disponer de un sistema de comunicación sofisticado que permita mantener conversaciones coherentes y productivas entre agentes. Estas conversaciones, que se dan siempre por iniciativa de los agentes implicados, marcan el carácter de la interacción entre ellos, la cual puede ser cooperativa o competitiva. Es importante destacar que entre agentes software no se realiza ninguna conexión restrictiva por lo que en ningún momento pierden su autonomía al sociabilizarse.

3.3.1 Diseño

La tecnología existente en la actualidad sobre agentes software es rica y variada [163] con gran cantidad de alternativas y metodologías de desarrollo. En cualquier caso el diseño de un sistema multiagente puede ser descendente (*top-down*) o ascendente (*bottom-up*). La elección de uno u otro depende principalmente de las restricciones del sistema.

El diseño descendente de sistemas multiagente se aplica cuando el elemento restrictivo es el entorno. Sus características quedan determinadas de antemano y son inflexibles. En este caso los agentes tienen como misión estudiarlo y controlarlo en la medida de sus posibilidades. Para ello extraen información de diferentes partes del mismo, la procesan, y actúan en consecuencia, directamente sobre su parte de entorno circundante o comunicándose con otros agentes vecinos. En el diseño descendente se suele tener como objetivo que los agentes no alteren el entorno más allá de sus acciones, es decir, que pasen lo más desapercibido posible. Su aplicación se centra principalmente en la resolución de problemas de sistemas distribuidos complejos en los que los agentes software son utilizados principalmente para simplificar la computación necesaria y para superar las limitaciones propias de las interfaces usuario-máquina [164].

El diseño ascendente de sistemas multiagente se aplica cuando el elemento restrictivo es el conjunto de agentes. En este caso sus características y su comportamiento quedan determinadas por adelantado y son un requisito indispensable. Este enfoque está dirigido a la creación de un sistema en el que observar cómo los agentes software evolucionan al convivir unos con otros formando una sociedad. Por lo tanto la aplicación de este tipo de diseño está especialmente indicado para elaborar modelos y simulaciones sociales [165] donde

los agentes representan a los individuos en cuestión, los cuales son definidos con todo detalle, y el sistema multiagente la propia sociedad que se pretende representar. Precisando más se puede decir que el ámbito de aplicación se corresponde con el modelado de sistemas adaptativos complejos, o *Complex Adaptive Systems* (CAS) [166]. Para sacar provecho de este modelado a base de agentes software debe existir una correspondencia directa de los mismos con los individuos reales en cuanto a sus comportamientos. Pueden adaptarse, cambiar, aprender, establecer relaciones dinámicamente entre ellos, formar organizaciones e interactuar a distancia. La evolución de la estructura de la población, de tamaño arbitrario, no es algo impuesto sino el elemento resultante del modelado [167].

Para el problema que trata este trabajo no se contempla un diseño ascendente del sistema multiagente. Aunque se ha aplicado con anterioridad para modelar redes guiadas [168] resulta mucho más complicado cuando éstas son redes inalámbricas por el característico comportamiento caótico de sus canales. Además no se busca un modelo de la red sino del enlace inalámbrico el cual no depende de los nodos-agentes sino del entorno. Por todo ello se descarta utilizar esta técnica.

El enfoque adoptado para el diseño del sistema multiagente implicado en el control del proxy buscado es el diseño descendente. Es el que mejor se ajusta a los objetivos que se desean alcanzar, los cuales requieren de mecanismos distribuidos para simplificar el proceso de control necesario con el que mitigar el problema de las interrupciones.

3.3.2 Aplicaciones

Tradicionalmente se han elaborado listas de actividades donde es posible aplicar agentes software. Esta acotación, que tiene como finalidad delimitar el ámbito de

esta tecnología, puede llevar a error por el carácter cerrado que imprime. Las categorías implicadas en estas listas no son campos de aplicación sino ejemplos concretos de aplicación. Los agentes software pueden usarse en cualquier actividad donde sea preciso sustituir total o parcialmente un grupo de actores autónomos capaces de comunicarse y tomar decisiones, generalmente usuarios de un sistema informático o miembros de una simulación. Los motivos para ello pueden ir desde simplemente reducir costes, mecanizando tareas sencillas y repetitivas, hasta alcanzar un rendimiento imposible para los humanos en determinadas labores.

Existe gran cantidad de ejemplos de uso de agentes software y continuamente van apareciendo nuevos y más variados. Por ejemplo en marketing y publicidad se han utilizado para llegar a un trato individualizado [169], lo cual requiere del manejo de gran cantidad de datos. Lo mismo ha ocurrido para la planificación y optimización de viajes [170] mediante procesos de minería de datos. La organización de librerías distribuidas digitales [171] es otro ejemplo donde es necesario procesar gran cantidad de información y los agentes software logran automatizar.

Otras aplicaciones más avanzadas son la asistencia personal en domótica por medio de inteligencia ambiental [172] o la dirección de proyectos de desarrollo de software [173]. En la educación también han llegado los agentes software para gestionar planes individualizados de aprendizaje [174]. En sanidad complementan a los médicos en el diagnóstico precoz y pronóstico de tumores cerebrales [175].

Los agentes software son especialmente potentes en el control de procesos. En transportes los agentes software han simplificado las tareas de control aéreo [176] aumentando la seguridad. En estaciones eólicas monitorizan y maximizan el rendimiento, todo ello en un entorno muy inestable como es el aire [177]. La

monitorización meteorológica también es otro ejemplo donde los agentes software se han incorporado para valorar la calidad del aire y vigilar continuamente los datos aportados por radares meteorológicos [178]. El caso de este trabajo donde los agentes monitorizan y controlan un enlace inalámbrico por donde se efectúa *streaming* entra también en este grupo de agentes software dedicados al control de procesos.

Aunque los agentes software se suelen emplear como solución a problemas hay casos donde el efecto conseguido es el contrario, llegando a ser hasta peligrosos. Los polémicos agentes de bolsa ultra rápidos [179] están convirtiendo el mundo de las finanzas en un campo de guerra encarnizada sin ningún control y con una inestabilidad que afecta directamente a la economía real y a la vida de las personas.

En redes se han utilizado también los agentes software. Para la administración de redes se han buscado agentes proactivos capaces de actuar antes de que los problemas ocurran y así poder evitarlos [180]. Otro caso son los agentes dedicados a la detección de intrusos con un comportamiento inspirado en el sistema inmunológico [181]. Los agentes móviles son especialmente útiles en redes heterogéneas donde por ejemplo éstos viajan por las mismas para mejorar su fiabilidad y QoS inspeccionando los diferentes tipos de tráfico que soportan [182]. Otra aplicación de este tipo de agentes es el encaminamiento en redes inalámbricas dinámicas [183].

En *streaming* con móviles se han utilizado agentes software para controlar la QoS monitorizando el canal para lograr distinguir la degradación de calidad producida por congestión de la red de la producida por errores del enlace inalámbrico [184]. Otro caso interesante es el de los agentes para la gestión

individual de los servicios proporcionados por un proveedor eligiendo la mejor estrategia de despliegue en cada caso según unos requerimientos específicos [185]. En este último ejemplo se utiliza la herramienta elegida en este trabajo para implementar agentes software: la plataforma *Java Agent DEvelopment Framework* (*JADE*) [186].

3.4 Toma de decisiones

La actividad fundamental de un agente software es la toma de decisiones. Para que se comporte inteligentemente debe ser capaz de tomar decisiones racionales. Una decisión racional es aquella con la que se consigue el mayor beneficio con el mínimo gasto de recursos, dependiendo del conocimiento que se tenga del entorno y los medios de acción disponibles.

Todos los seres vivos, desde los más simples a los más complejos, toman decisiones racionales continuamente. El beneficio buscado es la propia supervivencia y la de su legado genético. Su entorno cambia constantemente y esto supone una amenaza que debe contrarrestarse adecuando su comportamiento en cada momento. La vida en ningún caso transcurre de forma simple y estática sino muy al contrario, de forma extremadamente compleja y dinámica.

Las estrategias que siguen los seres vivos para tomar las decisiones que les permiten sobrevivir son muy variadas. Para alimentarse una bacteria puede asimilar partículas de su medio ambiente, algunas nocivas y otras nutritivas. Gracias a su composición y a las leyes físicas y químicas esas partículas son aceptadas o rechazadas de forma sistemática.

Los seres vivos con sistema nervioso pueden tomar decisiones más sofisticadas. Para ello tienen en cuenta su experiencia y las consecuencias que puede tener su comportamiento. Los animales más sencillos son eminentemente impulsivos en su forma de actuar: sólo consideran el momento presente. En cambio los animales más avanzados son capaces de considerar su futuro cercano y el conocimiento adquirido con anterioridad lo cual les permite enfrentarse a problemas más complicados con mejores resultados.

Los seres humanos son un caso extremo dentro del reino animal debido a su capacidad de abstracción superior. Con ello no sólo se han conseguido mejores decisiones, y por tanto una mayor garantía de supervivencia, sino además que las sociedades humanas sean las más avanzadas y complejas. La filosofía, la ciencia o el arte son algunos ejemplos de conocimiento generado por la relación de los seres humanos con el tiempo y su tendencia a abarcarlo. Sin embargo, este conocimiento aumenta cada vez más rápido llegando a comprometer las capacidades de sus creadores. Esto se debe a que este crecimiento tiene asociadas nuevas necesidades cada vez más complicadas de solventar. Por ello se ha tenido que investigar cómo mejorar los procesos de toma de decisiones ante las nuevas situaciones originadas. Con este panorama nace la teoría de la decisión.

La teoría de la decisión [187] es un área de estudio interdisciplinar que abarca tanto aspectos de la ciencia como de la psicología. Dependiendo de los objetivos que persigue se puede distinguir entre teoría de la decisión normativa y teoría de la decisión descriptiva. La teoría de la decisión normativa busca cómo tomar decisiones racionales y óptimas mediante métodos y técnicas que usan recursos matemáticos, técnicos y psicológicos. La teoría de la decisión descriptiva se dedica al estudio del comportamiento de los seres vivos, en especial los humanos, al tomar decisiones. Aunque son dos campos totalmente diferentes están íntimamente

relacionados entre sí y se retroalimentan propiciando así su avance mutuo. Este avance ha sido de gran ayuda para la resolución de decisiones en el peor escenario posible, la incertidumbre, donde no se tiene la información suficiente para valorar las consecuencias de las alternativas que se pueden llevar a cabo [188]. Temas destacados que han contribuido a ello dentro de la evolución de la teoría de la decisión [189] son entre otros el desarrollo del análisis de decisiones, la teoría de la utilidad subjetiva esperada en decisiones individuales, o la teoría de juegos en entornos con múltiples actores.

La inteligencia artificial, en su afán por construir sistemas artificiales capaces de comportarse de manera similar a los seres vivos y sobre todo a los humanos, ha hecho un uso extensivo de todo el conocimiento adquirido hasta el momento en lo referente a la teoría de la decisión. Sin embargo, la aplicación práctica de la teoría de la decisión no siempre es posible porque sus métodos están destinados a la búsqueda de la alternativa óptima y muchas veces esto no es posible, ya sea por escasez de recursos o falta de tiempo para la realización de todos los cálculos necesarios en el proceso. Por ello en inteligencia artificial es necesario buscar soluciones lo "suficientemente buenas" [190] más que las óptimas. En la práctica estas soluciones dependen de las limitaciones propias del contexto y de los márgenes predefinidos para que sean admitidas como validas.

Para que los agentes software puedan tomar decisiones autónomamente además de los métodos propios de la teoría de la decisión se hace uso de múltiples herramientas como son la lógica clásica, la lógica modal o la probabilidad [191]. También existen herramientas más especializadas y de nivel más alto [192], algunas de ellas inspiradas en procesos biológicos [193], que aunque no siguen de forma estricta el proceso estándar de toma de decisiones pueden ser utilizadas para tal fin con resultados satisfactorios.

A continuación se trata el proceso de decisión individual limitado a un único actor y algunas de las técnicas y herramientas que pueden utilizarse para su ejecución.

3.4.1 Proceso de decisión

El proceso de decisión es el conjunto de operaciones ordenadas dirigidas a la toma de las decisiones necesarias para solucionar un problema. Aunque existen varias versiones aquí se hará mención sólo de la más general dedicada a las decisiones individuales en las que interviene sólo un decisor [194]. Para su estudio es necesario la definición de sus elementos característicos:

- *Alternativa*: cada una de las posibles acciones que se pueden realizar, las cuales son excluyentes entre sí.
- *Consecuencia*: previsión del resultado de la adopción de una alternativa en un estado de la naturaleza concreto.
- *Estado de la naturaleza*: escenario posible o evento que influye en el proceso de decisión pero que no es controlable por el decisor.
- *Decisor*: actor encargado de realizar la elección de la alternativa más acertada según el criterio adoptado.
- *Criterio*: norma para valorar las alternativas y elegir la mejor de ellas.

Un proceso de decisión está condicionado por el conocimiento que se tenga del estado de la naturaleza, el cual puede ser total o parcial. El grado de este conocimiento determina el ambiente o contexto del proceso de decisión. De forma general se distinguen tres tipos de ambientes:

1. *Ambiente de certidumbre*: contexto en el que el decisor tiene un conocimiento total del estado de la naturaleza. De esta forma cada alternativa lleva a un resultado único y bien definido.

2. *Ambiente de riesgo*: contexto en el que el decisor se enfrenta a varios estados de la naturaleza posibles y conoce la probabilidad de cada uno.

3. *Ambiente de incertidumbre*: contexto en el que el decisor se enfrenta a varios estados de la naturaleza posibles y no conoce sus probabilidades.

Para que exista un proceso de decisión es necesario que existan dos o más alternativas. La adopción de una de ellas imposibilita la realización de cualquiera de las restantes. La motivación de este proceso es una situación ambigua que debe ser resuelta para la consecución de un fin determinado.

En todo proceso de decisión, ilustrado en la Fig. 26, se encuentran las siguientes fases:

1. *Análisis del problema*: estudio del problema que origina la necesidad de tomar una decisión.

2. *Generación de alternativas*: concepción de las posibles alternativas que pueden minimizar el problema.

3. *Evaluación de alternativas*: adjudicación de un grado de utilidad a cada alternativa en cada estado de la naturaleza basándose en la predicción de sus consecuencias.

4. *Elección de una alternativa*: selección y puesta en marcha de la alternativa más beneficiosa según un criterio de decisión predefinido.

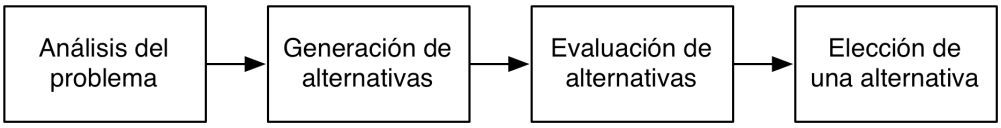

Fig. 26. Proceso de decisión

Estas fases son genéricas: para cada problema se tiene que diseñar la forma concreta de ejecución de cada una de ellas.

3.4.2 Técnicas y herramientas

Una decisión puede ser programada o no programada [195]. Las decisiones programadas abordan problemas que se presentan con cierta frecuencia y se resuelven con un conjunto de políticas o reglas preestablecidas. Las decisiones no programadas abordan problemas novedosos o inusuales y requieren un proceso especial en cada caso.

Para tomar decisiones no programadas es necesario adaptar el proceso de decisión genérico al problema concreto que se quiere minimizar. Esto requiere soluciones creativas, conocimiento y experiencia. Si el problema se repite con frecuencia es conveniente automatizar el proceso creado dando lugar a una decisión programada. Siempre existe el riesgo de que en cada iteración no se elija la alternativa óptima si las reglas seguidas no son las más adecuadas. En caso de llegar a un resultado final apropiado se consigue un ahorro de tiempo y de recursos considerable. Las herramientas básicas y más comunes para todo tipo de decisiones son las tablas y árboles de decisión. Para automatizar decisiones programadas se suelen emplear sistemas basados en reglas.

Una tabla de decisión es un medio de representación de una decisión elemental. En ella se enfrentan las diferentes alternativas con los posibles estados de la naturaleza que se pueden dar. Cada celda se corresponde con una consecuencia e indica su valoración. A partir de esta representación se puede evaluar cada alternativa para finalmente elegir la más beneficiosa. En un ambiente de certidumbre el valor de cada alternativa concuerda con el de su única consecuencia posible. En ambientes de riesgo o incertidumbre donde cada alternativa puede tener varias consecuencias posibles este valor se calcula aplicando un criterio. En ambiente de riesgo se suelen emplear criterios como el del valor esperado, el de mínima varianza con media acotada, el de la media con varianza acotada, el de dispersión o el de la probabilidad máxima, y en ambiente de incertidumbre el de Wald, el Maximax, el de Hurwicz, el de Laplace o el de Savage.

Los árboles de decisión son más potentes y flexibles que las tablas de decisión bidimensionales. Permiten representar en un mismo gráfico decisiones complejas formadas por varias decisiones elementales relacionadas cronológicamente. En ellos los enlaces simbolizan las alternativas y los estados de la naturaleza. Cada nodo se corresponde con una decisión si de él parten varias alternativas o con un resultado incierto si de él parten diferentes estados de la naturaleza. Los nodos hoja coinciden con las consecuencias finales. Para determinar las mejores alternativas se siguen los mismos criterios empleados en las tablas de decisión con la diferencia de que las valoraciones de las consecuencias pueden venir dadas o ser el resultado de decisiones posteriores.

La Fig. 27 y la Fig. 28 muestran dos ejemplos básicos de tabla y árbol de decisión respectivamente. Modelan el mismo problema con dos estados de la naturaleza e1 y e2, dos alternativas a1 y a2, y sus respectivas consecuencias c11, c12, c21 y c22.

	e1	e2
a1	c11	c12
a2	c21	c22

Fig. 27. Tabla de decisión

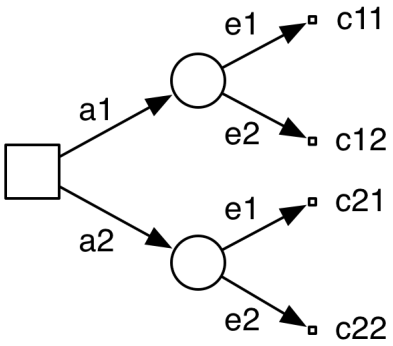

Fig. 28. Árbol de decisión

Los sistemas basados en reglas [196] son herramientas que implementan razonamiento por defecto, razonamiento abductivo y revisión de creencias. Para ello emplean la lógica no monotónica. Esta permite que el conocimiento adquirido pueda variar libremente, aumentando o disminuyendo, al agregar conocimiento nuevo. Estos sistemas trabajan directamente con una representación de hechos sobre los que aplican reglas, conocidas como reglas de producción e introducidas formalmente por Emil L. Post en los años 1940 [197]. Éstas son implicaciones que establecen relaciones antecedente-consecuente de la forma "SI condición ENTONCES acción" entendiéndose como condición una expresión lógica y como acción la modificación del conjunto de hechos u otra operación externa al sistema.

El razonamiento por defecto permite obtener conclusiones a partir de un conocimiento incompleto si éstas no producen contradicciones. El razonamiento abductivo proporciona a partir de un hecho sus posibles causas según el conocimiento adquirido. La revisión de creencias se encarga de mantener constantemente la consistencia del conocimiento retrayendo aquellos hechos que se han vuelto contradictorios en un momento dado. El razonamiento por defecto se realiza mediante encadenamiento de reglas hacia delante y el razonamiento abductivo mediante encadenamiento de reglas hacia atrás. En cada iteración se

determinan las reglas que pueden ser disparadas y se selecciona una de ellas en base a un criterio preestablecido. La ejecución se suspende cuando no hay reglas combinables con los hechos disponibles.

3.5 Predicción de estados

En el mundo real una toma de decisiones suele estar asociada a un número elevado de estados de la naturaleza. Para simplificar una decisión reduciendo esta cantidad es necesario detectar aquellos estados que son verdaderamente improbables y descartarlos. Esto requiere de predicciones que sean capaces de proporcionar resultados fiables a partir de información incompleta, voluble e imprecisa.

Entendemos como predicción la estimación del estado siguiente $\hat{s}_{n+1}$ a la secuencia de estados $\{s_n\}$ recopilada hasta el momento presente.

$$\{s_n\} = s_1, s_2, \ldots, s_n$$

La predicción es función F de la secuencia de estados.

$$\hat{s}_{n+1} = F(\{s_n\})$$

El error cometido e es proporcional a la distancia d entre la predicción $\hat{s}_{n+1}$ y el nuevo estado s_{n+1}.

$$e \propto d(\hat{s}_{n+1}, s_{n+1})$$

Consideramos que minimizando este error se pueden tomar mejores decisiones para gestionar las interrupciones de vídeo-streaming maximizando así la QoE.

$$\min(e) \Rightarrow \max(QoE)$$

El problema a solucionar es la determinación de la función F que minimice el error e y que sea implementable en tiempo real firme t (menos de 1 segundo) y con un consumo de recursos computacionales C mínimo. Este es un problema de optimización matemática que debe dar resultados correctos en tiempo real.

$$F \Rightarrow \min(e) \wedge \min(t) < 1 \wedge \min(C)$$

Para predecir interrupciones de video-streaming hemos descartado el uso de un modelo matemático analítico del enlace inalámbrico y también su simulación por medio de herramientas avanzadas como un sistema multiagente. A continuación analizamos una serie de métodos matemáticos muy utilizados en control inteligente y candidatos a implementar la función F, al ser aplicables en la predicción a corto plazo del comportamiento de sistemas no lineales caóticos deterministas. Éstos son: redes neuronales artificiales, algoritmos genéticos y lógica difusa.

3.5.1 Redes neuronales artificiales

Las redes neuronales artificiales son modelos computacionales simplificados de redes neuronales biológicas. Originalmente fueron planteadas para el estudio y simulación del sistema nervioso central en áreas como la neurociencia teórica [198]. Algunas de sus cualidades, como la flexibilidad, robustez, complejidad, paralelismo, plasticidad o adaptabilidad, han propiciado que también se usen para el procesamiento automático de datos [199]. Diferentes variaciones de las redes neuronales artificiales se emplean en problemas esencialmente de control, clasificación y reconocimiento de patrones en los que no es posible adoptar una solución basada en un modelo matemático preciso y de complejidad manejable.

Una red neuronal está formada por un cúmulo de neuronas interconectadas que colaboran para producir un conjunto de respuestas a partir de un conjunto de estímulos. Cada neurona está compuesta básicamente por un cuerpo (soma) con varios receptores (dendritas) de impulsos nerviosos externos y un único emisor (axón) del impulso nervioso disparado por la neurona. En cada conexión (sinapsis) entre neuronas la emisora (neurona presináptica) influye en las receptora (neurona postsináptica) excitándola o inhibiéndola. Una neurona mantiene una diferencia de potencial (potencial de reposo) entre el interior y el exterior de su cuerpo. Cuando recibe impulsos nerviosos externos esa diferencia cambia dando lugar a un nuevo potencial (potencial postsináptico). Si éste supera cierto umbral la neurona termina disparando un impulso nervioso (potencial de acción).

Una neurona artificial con índice k está constituida por tres elementos:

1. *Conjunto de sinapsis*. Cada sinapsis queda definida por un peso sináptico w_{kj} en el que j es el índice de la neurona presináptica. La magnitud de este peso expresa la intensidad de la sinapsis y su signo el carácter excitatorio o inhibitorio.

2. *Función de excitación* σ_k. Esta función asocia un potencial postsináptico v_k al par formado por los pesos sinápticos $\{w_{kj}\}$ y los potenciales de acción de las neuronas presinápticas $\{a_j\}$.

$$v_k = \sigma_k\left(\{w_{kj}\},\{a_j\}\right)$$

3. *Función de activación* φ_k. Esta función asocia un potencial de acción a_k al potencial postsináptico v_k. Básicamente limita su amplitud acotándolo en un intervalo.

$$a_k = \varphi_k(v_k)$$

La función de excitación más común es la suma ponderada. A ésta se añade un sesgo b_k como potencial de reposo y un sesgo c_k como potencial de acción inducido por un estímulo. En una red neuronal artificial con q neuronas se expresa de la siguiente manera:

$$\sigma_k\left(\{w_{kj}\},\{a_j\}\right) = \sum_{j=1}^{q} w_{kj} a_j + b_k + c_k$$

Las funciones de activación más frecuentes son:

- *Función escalón*: $\varphi_k(v_k) = \begin{cases} 1, \ si \ \ v_k > 0 \\ 0, \ si \ \ v_k \le 0 \end{cases}$
- *Función sigmoide*: $\varphi_k(v_k) = \dfrac{1}{1+e^{-v_k}}$, $0 \le \varphi_k(v_k) \le 1$
- *Función tangente hiperbólica*: $\varphi_k(v_k) = \tanh(v_k)$, $-1 \le \varphi_k(v_k) \le 1$

La Fig. 29 muestra un esquema gráfico de una neurona artificial.

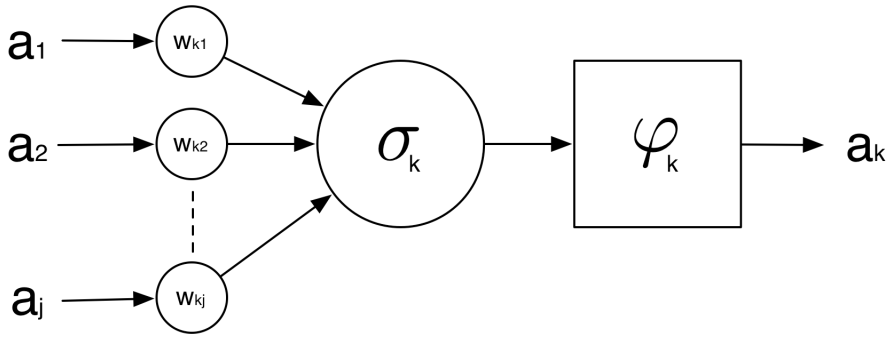

Fig. 29. Neurona artificial

En una red neuronal artificial las neuronas se organizan en niveles. Se distinguen tres tipos: de entrada, de salida y ocultos. El nivel de entrada está formado por las neuronas que reciben los estímulos y el nivel de salida por las que proporcionan las respuestas de la red. Los niveles ocultos están compuestos por el resto de neuronas y se sitúan entre el nivel de entrada y el de salida. Según la orientación de las sinapsis entre niveles se distinguen redes de propagación hacia delante si todas las sinapsis van en sentido entrada-salida y redes recurrentes si por lo menos una de las sinapsis forma un ciclo en su nivel o entre niveles diferentes.

El aprendizaje en redes neuronales artificiales es un proceso iterativo a modo de entrenamiento en el que se ajustan los pesos sinápticos w_{kj} de todas las neuronas para mejorar progresivamente las respuestas de la red (valores a_k de las neuronas del nivel de salida) según un criterio. Existen varios paradigmas de aprendizaje:

1. *Aprendizaje supervisado.* En este tipo de aprendizaje se minimiza una función de error de las respuestas, generalmente la distancia entre las que proporciona la red y otras objetivo que son conocidas y consideradas correctas. El aprendizaje supervisado se emplea sobre todo en reconocimiento de patrones preestablecidos y en aproximación de funciones.

2. *Aprendizaje no supervisado.* En este tipo de aprendizaje se minimiza una función de costo aplicada a la relación estímulos-respuestas. Se emplea principalmente en agrupaciones (*clustering*).

3. *Aprendizaje por refuerzo.* En este tipo de aprendizaje por cada respuesta se obtiene una valoración o recompensa. Se busca maximizar la recompensa acumulada por la secuencia de respuestas obtenidas en un

intervalo de tiempo. Se emplea especialmente en problemas de control automatizado.

La duración del aprendizaje en redes neuronales es variable y no puede garantizarse que finalice y llegue a un resultado satisfactorio en todos los casos. Normalmente se establece un número máximo de iteraciones para prevenir que este proceso se eternice.

Para aproximar la función F una posibilidad es emplear una red neuronal artificial con aprendizaje supervisado. En este caso los estímulos de esta red se corresponden con la secuencia de estados $\{s_n\}$ y su única respuesta con la estimación del estado siguiente $\hat{s}_{n+1}$. La respuesta objetivo para el aprendizaje es el estado siguiente s_{n+1}.

Existen varios modelos de redes neuronales artificiales que han sido utilizados satisfactoriamente en problemas de predicción del comportamiento de sistemas caóticos deterministas. En este caso son destacables los buenos resultados de las redes de propagación hacia delante como el perceptrón multinivel [200] por ser las más simples. Para este problema las redes recurrentes y especialmente la basada en el modelo no lineal autorregresivo con entradas exógenas ($NARX$) [201] suelen proporcionar mejores respuestas que el perceptrón multinivel pero a costa de un mayor consumo de recursos computacionales y de tiempo, el cual puede llegar a ser muy elevado. Consideramos que es conveniente valorar primero el perceptrón multinivel antes de continuar con modelos más complejos.

Para optimizar la arquitectura del perceptrón multinivel buscando mejores respuestas ($min(e)$) y una cantidad reducida de neuronas ($min(t) \wedge min(C)$) se debe minimizar el tamaño del nivel de entrada, considerando sólo una ventana deslizante con los últimos m estados [202], y minimizar la cantidad y tamaño de

los niveles ocultos [203]. La expresión de esta red, con q neuronas numeradas siendo las m primeras las de entrada y la última la de salida y siendo a_k^n y c_k^n los valores a_k y c_k en la iteración n, es la siguiente:

$$c_k^n = \begin{cases} \tau\left(s_{n-m+k}\right) & , \; si \;\; 1 \le k \le m \\ 0 & , \; si \;\; m < k \le q \end{cases}$$

$$a_k^n = \varphi_k\left(\sum_{j=1}^{q} w_{kj} a_j^{n-1} + b_k + c_k^n\right)$$

$$\hat{s}_{n+1} = \tau^{-1}\left(a_q^n\right)$$

La función invertible τ permite operar directamente con estados en la red neuronal artificial. Esta función asocia a cada estado s un potencial de acción a y viceversa de la siguiente manera:

$$a = \tau(s)$$
$$s = \tau^{-1}(a)$$

Los pesos sinápticos w_{kj} forman parte de la matriz de pesos sinápticos $\mathbf{W} \in \mathcal{M}_{q\times q}(\mathbb{R})$. Esta matriz cuadrada define la estructura de la red. La de un perceptrón multinivel tiene una forma característica de bloques en escalera descendente con hueco inferior. Siempre es una matriz triangular inferior en la que los elementos de la diagonal principal son nulos. Esto es debido a que en un perceptrón multinivel sólo existe sinapsis si la neurona presináptica se encuentra en el nivel inmediato anterior al de la postsináptica. Cada bloque-escalón de la escalera resultante se corresponde con las sinapsis existentes entre dos niveles contiguos. La matriz $\mathbf{W}$ de un perceptrón multinivel con un solo nivel oculto es la siguiente:

$$\mathbf{W} = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 \\ w_{m+1,1} & w_{m+1,2} & \cdots & w_{m+1,m} & 0 & 0 & \cdots & 0 & 0 \\ w_{m+2,1} & w_{m+2,2} & \cdots & w_{m+2,m} & 0 & 0 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ w_{q-1,1} & w_{q-1,2} & \cdots & w_{q-1,m} & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & \cdots & 0 & w_{q,m+1} & w_{q,m+2} & \cdots & w_{q,q-1} & 0 \end{bmatrix}$$

El número ω de sinapsis de un perceptrón multinivel es igual al sumatorio del producto del número de neuronas de cada par de niveles contiguos. En este caso para un perceptrón multinivel con un solo nivel oculto, siendo q_E el número de neuronas del nivel de entrada, q_O el número de neuronas del nivel oculto y q_S el número de neuronas del nivel de salida, el número de sinapsis es el siguiente:

$$\omega = q_E q_O + q_O q_S = q_O\left(q_E + q_S\right)$$

Según esta relación y suponiendo que siempre $q_O \propto q_E$ se deduce que el número de sinapsis crece cuadráticamente con respecto a q_E.

$$\omega \propto {q_E}^2 + q_E q_S$$

En este caso $q_E = m$ y $q_S = 1$ por lo que:

$$\omega \propto m^2 + m$$

En la Fig. 30 se muestra un gráfico representativo del perceptrón multinivel con un solo nivel oculto descrito. Las neuronas se corresponden con los círculos y las sinapsis con las líneas.

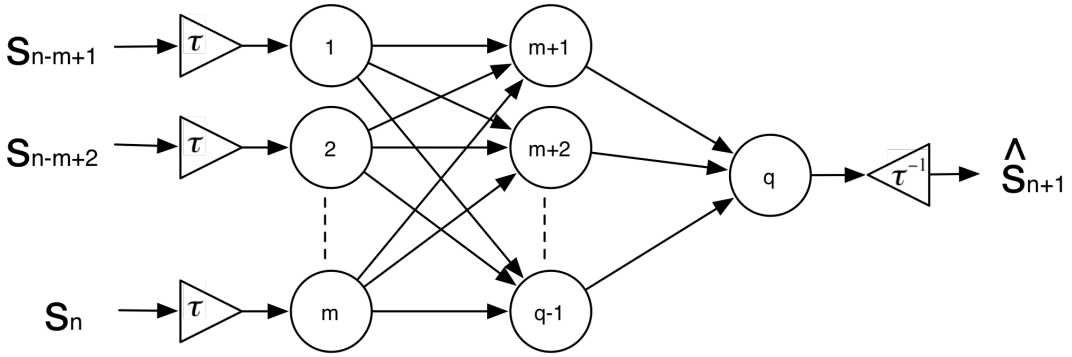

Fig. 30. Red neuronal artificial

El aprendizaje en esta red neuronal artificial está especialmente condicionado por tres factores [204]:

1. *Algoritmo de aprendizaje supervisado.* Según el tipo de algoritmo empleado la convergencia es más o menos rápida. Su eficiencia es vital en este caso por la necesidad de entrenar in situ la red, simultaneando ejecución y aprendizaje, al ser muy elevada la variabilidad del estado del enlace inalámbrico y su entorno. Pequeños cambios en la localización espacio-temporal del dispositivo móvil pueden invalidar todo lo aprendido hasta el momento obligando a readaptar la red continuamente.

2. *Arquitectura de la red.* Los dos problemas principales relacionados con el aprendizaje y la arquitectura de la red son la ambigüedad y el sobreentrenamiento. La ambigüedad se produce cuando son válidas diferentes respuestas para una misma combinación de estímulos. Esta situación sucede si m es muy pequeño. El sobreentrenamiento es un fenómeno que se da cuando la red se ajusta demasiado a los datos utilizados en el aprendizaje perdiendo la capacidad de generalizar y de responder ante nuevos estímulos. Los motivos suelen ser demasiadas neuronas en niveles ocultos y pocos datos dedicados al aprendizaje. Esta

situación se cumple si m es muy elevado. A mayor m mayor complejidad de la función aproximada y mayor cantidad de puntos necesarios para definir su curva. El sobreentrenamiento es especialmente problemático cuando ocurren sucesos esporádicos relevantes que se salen de la norma como son las interrupciones.

3. *Calidad de la muestra de entrenamiento.* La muestra para aproximar una función debe ser lo menos ruidosa posible y contener una cantidad de puntos suficiente para definir la curva de la función. Los puntos más relevantes son los máximos y mínimos locales y los puntos de inflexión. Los demás puntos son necesarios para definir con precisión las concavidades y convexidades de la curva. La concentración de puntos en determinadas zonas dejando otras vacías es un problema que reduce la calidad de la muestra y que para solventarlo es necesario esperar a la incorporación de nuevos puntos. Durante este tiempo la red no se adapta convenientemente quedando inoperativa, lo cual es inadmisible si éste es elevado.

Todas estas dificultades invitan a considerar otros métodos adaptativos, con una arquitectura menos dependiente del valor de m y un tiempo de espera reducido para obtener predicciones fiables.

3.5.2 Algoritmos genéticos

Los algoritmos genéticos son un método de búsqueda y optimización matemática inspirado en la genética y la teoría de la evolución. Fueron formalizados y empezaron a popularizarse en la década de 1970, destacando en esta época por su influencia los trabajos publicados por John H. Holland [205].

En los algoritmos genéticos se hace evolucionar una población inicial de soluciones (individuos) las cuales van mejorando progresivamente tras sucesivas iteraciones (generaciones). Se considera cada solución como la manifestación (fenotipo) de un conjunto (genotipo) de propiedades modificables (genes), el cual representa a la solución en el algoritmo. En cada iteración cada solución es evaluada y las menos aptas son eliminadas. Las restantes se reproducen experimentando combinación de soluciones (recombinación genética) o alteraciones espontáneas (mutaciones). Así la nueva población tiende a ser mejor que su predecesora. Este elitismo hace que al aumentar el número de iteraciones también aumente la probabilidad de que aparezca finalmente una solución óptima.

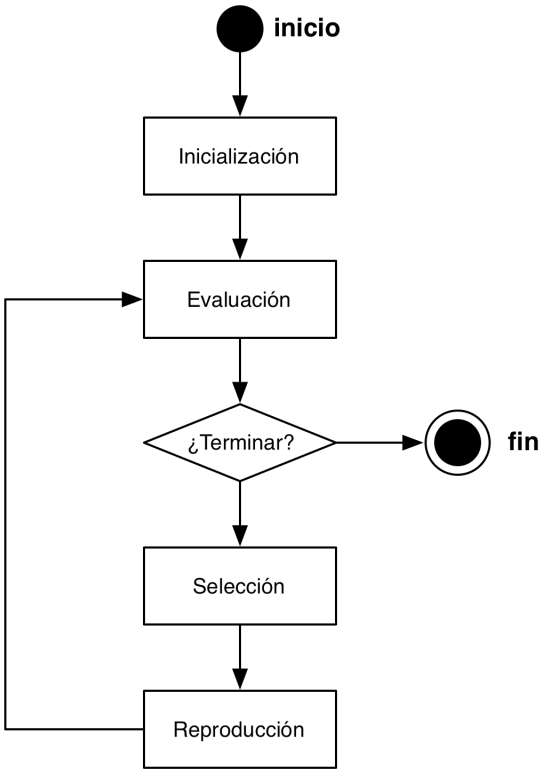

Fig. 31. Algoritmo genético

Los algoritmos genéticos pueden adoptar diversas formas por lo que no hay un patrón estricto. Éstas dependen principalmente de cómo se realice la selección o la reproducción. En general un algoritmo genético, ilustrado en la Fig. 31, consta de las siguientes fases:

1. *Inicialización*. Creación de una población inicial de n genotipos $\{x_n\}$ de soluciones. Se puede realizar aleatoriamente garantizando una diversidad representativa de todo el conjunto de genotipos posibles para evitar una convergencia prematura del algoritmo genético en un óptimo local.

$$\{x_n\} = x_1, x_2, ..., x_n$$

2. *Evaluación*. Aplicación de la función de aptitud f a cada genotipo x_i la cual proporciona su medida de aptitud a_i. Al finalizar la evaluación se considera si terminar la ejecución del algoritmo. Para ello se siguen criterios como el haber superado un número máximo de iteraciones, cumplir ciertas expectativas o el no existir cambios en la población en iteraciones consecutivas lo cual es síntoma de haber alcanzado un óptimo.

$$a_i = f(x_i) \ , \ 1 \leq i \leq n$$

3. *Selección*. Eliminación de los genotipos menos aptos. Éstos son los que no superan un cierto umbral de aptitud. Los supervivientes conforman la población destinada a reproducirse y tener descendencia.

4. *Reproducción*. Creación de una nueva población a partir de la anterior previamente seleccionada. La reproducción se realiza aplicando operadores genéticos como la recombinación y la mutación. Con la recombinación se obtiene un nuevo genotipo resultado de la unión de sus padres. Este

operador se corresponde con la reproducción sexual. La mutación produce modificaciones al azar de uno o varios genes de un genotipo. Este operador permite alcanzar zonas del espacio de búsqueda inicialmente inaccesibles si sólo se aplica la recombinación. Una vez finalizada esta fase se vuelve a la de "*Evaluación*".

Las partes más relevantes del diseño de un algoritmo genético son la determinación de la estructura de los genotipos y el establecimiento de la función de aptitud. De ellas depende la velocidad de convergencia. Si ésta es muy lenta el algoritmo deja de ser válido en aplicaciones con restricciones temporales. Si es muy rápida el algoritmo tiende a converger sólo hacia óptimos locales. Esto suele suceder cuando aparece con facilidad un superindividuo que termina formando una plaga acabando con la variabilidad genética y la posibilidad de alcanzar el óptimo global.

Una posibilidad para aproximar la función F es el empleo de un algoritmo genético. En este caso los genotipos se corresponden con expresiones analíticas de funciones en las que se combinan componentes de la secuencia de estados $\{s_n\}$ con operadores matemáticos. La aplicación de cada una de estas funciones proporciona la estimación del estado siguiente $\hat{s}_{n+1}$. La medida de aptitud de cada genotipo debe estar relacionada con el error cometido en cada estimación con respecto al valor real del estado siguiente s_{n+1}. A menor error mayor aptitud y viceversa.

Los algoritmos genéticos han demostrado su capacidad para resolver problemas de predicción de sistemas caóticos deterministas reales [206]. Aunque en muchos de estos problemas es posible emplear otros métodos los algoritmos genéticos tienen como ventaja frente a ellos el proporcionar explícitamente una expresión analítica aproximada de la evolución dinámica del sistema [207]. La ejecución de una expresión matemática simple normalmente utiliza poco tiempo y

recursos. Esta expresión también resulta interesante para seguir investigando el problema.

Aunque un algoritmo genético puede dar buenos resultados, en sistemas muy dinámicos el tiempo de convergencia aumenta incumpliéndose el requisito de eficiencia ($min(t) \wedge min(C)$) y puede no alcanzarse incumpliéndose además el requisito de eficacia ($min(e)$). En algunos casos es posible reducir estos inconvenientes delimitando el espacio de búsqueda si se dispone de un modelo aproximado del problema, es decir, si se conoce previamente la forma de la expresión que se desea obtener [208]. En este caso el algoritmo genético en vez de generar expresiones completas se limita a ajustar parámetros.

Para los casos en los que no se dispone de un modelo del problema otra alternativa interesante consiste en combinar un algoritmo genético con algún modelo simple de proceso estocástico como el modelo autorregresivo (AR) [209]. En éste un estado es una combinación lineal de sus estados anteriores. El modelo AR(m) de orden m, siendo c una constante, θ_i cada uno de sus parámetros y ε_n el residuo (ruido blanco), se define como:

$$s_n = c + \sum_{i=1}^{m} \theta_i s_{n-i} + \varepsilon_n$$

Dada la secuencia de estados se ajustan los parámetros minimizando el residuo. Una vez calculados estos parámetros la expresión de la estimación del estado siguiente $\hat{s}_{n+1}$ se define como:

$$\hat{s}_{n+1} = c + \sum_{i=1}^{m} \theta_i s_{n+1-i}$$

Los parámetros θ_i del modelo forman la matriz de parámetros $\boldsymbol{\Theta} \in \mathcal{M}_{1\times m}(\mathbb{R})$. Ésta es una matriz densa de m elementos y su expresión es la siguiente:

$$\boldsymbol{\Theta} = \begin{bmatrix} \theta_1 & \theta_2 & \cdots & \theta_m \end{bmatrix}$$

Evidentemente el modelo AR(m) es lineal y no puede aproximar la función F al completo pero sí por partes. El algoritmo genético genera una población de modelos AR(m) que va evolucionando y selecciona en cada momento aquel que proporciona estimaciones del estado siguiente con menor error. Estos modelos además de parámetros diferentes pueden tener valores de m distintos.

Emplear modelos AR(m) es una simplificación que reduce el tamaño del espacio de búsqueda del algoritmo genético y también su tiempo de convergencia, lo que aumenta su eficiencia. Asimismo la muestra de puntos de la función F necesaria para ajustar un modelo AR(m) que la aproxime parcialmente es mucho más pequeña, lo que consecuentemente disminuye el tiempo de espera por esa muestra. No obstante todo esto sólo es efectivo si se mantiene una diversidad de modelos elevada. Si el algoritmo genético converge en sólo uno y el resto desaparece no podrá adaptarse o lo hará muy lentamente cuando las características del estado del canal inalámbrico cambien. Esta situación es inadmisible.

Otra situación problemática es la aparición de los típicos saltos bruscos en el estado del canal inalámbrico. Un modelo AR(m) es incapaz de predecir estos saltos porque sólo es realmente eficaz cuando el estado del canal inalámbrico es estacionario. Solamente después de que ocurran estos saltos el algoritmo genético logra reajustar los parámetros del modelo restituyendo la efectividad perdida. Un método aceptable debe predecir estos saltos bruscos reconociendo los patrones de estados que se dan con anterioridad a ellos.

3.5.3 Sistemas de lógica difusa

La lógica difusa es una lógica no-clásica plurivalente que admite infinitos valores de verdad entre falso y verdadero y rechaza el principio del tercero excluido por lo que en ella la disyunción de una proposición y su negación $(p \vee \neg p)$ no es siempre verdadera. Esta lógica se suele emplear para modelar sistemas complejos no lineales cuyas características no son categorizables con precisión. Aparece en la década de 1960 con la publicación de la teoría de conjuntos difusos desarrollada por Lofti A. Zadeh [210].

Un conjunto difuso es un conjunto con relación de pertenencia gradual. Formalmente un conjunto difuso A en U siendo U el universo del discurso es un conjunto de pares ordenados:

$$A = \left\{ \left(x, \mu_A(x)\right) \mid x \in U \right\}$$

La función μ_A conocida como función de pertenencia asocia a cada elemento $x \in U$ un grado de pertenencia al conjunto difuso A. El valor del grado de pertenencia varía entre 0 (pertenencia nula) y 1 (pertenencia total). La función de pertenencia se define como:

$$\mu_A : X \rightarrow [0,1]$$

Como función de pertenencia se suele emplear la función triangular, la trapezoidal o la gaussiana acotadas en el intervalo $[0,1]$.

La función de pertenencia define la forma de un conjunto difuso lo que permite compararlo con otros. De este modo dos conjuntos difusos A en U y B en U se consideran iguales si y sólo si sus funciones de pertenencia también son iguales.

$$A = B \Leftrightarrow \mu_A(x) = \mu_B(x), \forall x \in U$$

Un conjunto difuso A en U está contenido en el conjunto difuso B en U si y sólo si la función de permanencia del primero es siempre menor o igual que la del segundo.

$$A \subseteq B \Leftrightarrow \mu_A(x) \leq \mu_B(x), \forall x \in U$$

Las operaciones básicas con conjuntos difusos son la intersección, la unión y el complemento. Éstas son similares a las operaciones con conjuntos convencionales. La intersección y la unión de conjuntos difusos cumplen con las mismas propiedades que sus homólogas de conjuntos convencionales: asociativa, conmutativa, distributiva, idempotencia, elemento neutro y elemento absorvente. En cambio el complemento de un conjunto difusos sólo cumple con la propiedad involutiva y las leyes de Morgan. La unión de un conjunto difuso y su complementario no siempre equivale al universo del discurso y la intersección de un un conjunto difuso y su complementario no siempre equivale al conjunto vacío.

La función de pertenencia de la intersección de dos conjuntos difusos A y B es el resultado de una operación binaria tipo norma triangular o t-norma. La más usada es la del mínimo.

$$\mu_{A \cap B}(x) = \min\left(\mu_A(x), \mu_B(x)\right)$$

La función de pertenencia de la unión de dos conjuntos difusos A y B es el resultado de una operación binaria tipo conorma triangular o t-conorma. La más usada es la del máximo.

$$\mu_{A \cup B}(x) = \max\left(\mu_A(x), \mu_B(x)\right)$$

La función característica del complemento de un conjunto difuso A es el resultado de una operación unaria tipo negación fuerte. La más usada es la negación usual.

$$\mu_{A^C}(x) = 1 - \mu_A(x)$$

La lógica difusa se fundamenta en la teoría de conjuntos difusos. Entre la teoría de conjuntos y la lógica proposicional existe una relación de isomorfismo: cada teorema enunciado en una tiene su homólogo en la otra. Las correspondencias básicas entre ellas son la de los conceptos de pertenencia de un elemento a un conjunto $(\_ \in \_)$ y de atribución lógica de una propiedad a un sujeto $(\_ \; es \; \_)$ y la de los operadores de teoría de conjuntos intersección $\cap$, unión $\cup$ o complemento ^{C} con los operadores lógicos conjunción $\wedge$, disyunción $\vee$ y negación $\neg$ respectivamente. Entre la teoría de conjuntos difusos y la lógica difusa existe la misma relación de isomorfismo.

Las expresiones de lógica difusa son similares a las de lógica proposicional. La notación no varía, sólo su interpretación. Las propiedades en lógica difusa son citadas usando términos lingüísticos (valores cualitativos) definidos matemáticamente por conjuntos difusos en un universo del discurso formado por elementos numéricos (valores cuantitativos). En este caso los conjuntos difusos sirven como puente entre la ambigüedad del lenguaje natural y la precisión de los formalismos matemáticos.

Un sistema de lógica difusa emplea reglas de lógica difusa para asociar a un conjunto X de valores numéricos de entrada un conjunto Y de valores numéricos de salida. Estas reglas son implicaciones lógicas del tipo: "*SI* antecedente *ENTONCES* consecuente". Para poder aplicarlas es necesaria una conversión previa de los elementos numéricos de X a elementos de los conjuntos difusos correspondientes

con los términos lingüísticos empleados en ellas. Esta conversión c se denomina *fuzzificación*:

$$\begin{aligned} c : X &\rightarrow A \\ x &\rightarrow x' = \left(x, \mu_A(x)\right) \end{aligned}$$

La aplicación de las reglas del sistema produce un nuevo conjunto difuso B por cada valor de salida. La conversión d de la función de pertenencia μ_B de cada conjunto difuso resultante a un valor numérico de Y se denomina *defuzzificación*:

$$\begin{aligned} d : \left(Y \rightarrow [0,1]\right) &\rightarrow Y \\ \mu_B \qquad &\rightarrow y \end{aligned}$$

Para llevar a acabo todas estas operaciones un sistema de lógica difusa posee cuatro componentes básicos:

1. *Fuzzificador*: calcula el grado de pertenencia de los valores numéricos de entrada a los conjuntos difusos que se corresponden con los términos lingüísticos de las reglas.

2. *Base de conocimiento*: almacena la definición de los términos lingüísticos y las reglas del sistema de lógica difusa.

3. *Motor de inferencia*: obtiene conclusiones a partir de la información proporcionada por el fuzzificador y las reglas de la base de conocimiento. Estas conclusiones son nuevos conjuntos difusos.

4. *Defuzzificador*: calcula los valores numéricos de salida a partir de los conjuntos difusos originados por el motor de inferencia.

Existen varios modelos de sistemas de lógica difusa. El más conocido y usado es el Mamdani [211] seguido del Sugeno [212]. Las diferencias entre estos modelos se centran especialmente en la expresión de las reglas y en el defuzzificador.

En el modelo Mamdani las reglas son puramente lingüísticas, tanto antecedente como consecuente. En este modelo, siendo q el número total de reglas, A_i^k los conjuntos difusos en $U_i \subseteq \mathbb{R}$, B^k el conjunto difuso en $V \subseteq \mathbb{R}$, $x_i \in U_i$ las variables numéricas de entrada e $y \in V$ la variable numérica de salida, una regla R^k con índice $k \leq q$ tiene la forma:

$$R^k : \; SI \;\; x_1 \;\; es \;\; A_1^k \;\wedge\; x_2 \;\; es \;\; A_2^k \wedge \ldots \wedge \;\; x_m \;\; es \;\; A_m^k \;\; ENTONCES \;\; y \;\; es \;\; B^k$$

La aplicación de la regla R^k comienza con el cálculo del valor w_k de su antecedente. En este caso al tratarse éste de una conjunción se emplea una operación t-norma. Si elegimos la del mínimo su expresión es la siguiente:

$$w_k = \min\left(\mu_{A_1^k}(x_1) \;,\; \mu_{A_2^k}(x_2) \;,\; \ldots \;,\; \mu_{A_m^k}(x_m)\right)$$

El resultado de la implicación entre antecedente y consecuente es un nuevo conjunto difuso B_*^k y éste se calcula con una operación t-norma. Las más habituales para este fin son la del mínimo y la del producto. Si elegimos la del mínimo su expresión es la siguiente:

$$\mu_{B_*^k}(y_*) = \min\left(w_k \;,\; \mu_{B^k}(y_*)\right)$$

Varias reglas pueden producir nuevos conjuntos difusos para una misma variable de salida. Es necesario agregar estas reglas para obtener un conjunto B' único. Esta agregación es una unión de conjuntos y para ella se emplea una operación t-conorma. La más frecuente en este caso es la del máximo. Si la elegimos su expresión es la siguiente:

$$\mu_{B'}(y_*) = \max\left(\mu_{B_*^1}(y_*)\ ,\ \mu_{B_*^2}(y_*)\ ,\ \ldots\ ,\ \mu_{B_*^q}(y_*)\right)$$

Finalmente se obtiene cada valor numérico de salida y a partir de su correspondiente conjunto difuso agregado. Para ello el método más usado es el cálculo del centroide o centro de gravedad del área descrita por la función de pertenencia del conjunto difuso agregado. Aunque su cálculo puede ser complicado garantiza una solución única. Su expresión es la siguiente:

$$y = \frac{\int y_* \mu_{B'}(y_*)\ dy_*}{\int \mu_{B'}(y_*)\ dy_*}$$

En el modelo Sugeno el antecedente de las reglas es lingüístico y el consecuente numérico. En este modelo, siendo q el número total de reglas, los A_i^k los conjuntos difusos en $U_i \subseteq \mathbb{R}$, los $x_i \in U_i$ las variables numéricas de entrada, $y \in \mathbb{R}$ la variable numérica de salida y $f^k : \mathbb{R}^m \to \mathbb{R}$ una función matemática, una regla R^k con índice $k \leq q$ tiene la forma:

$$R^k:\ SI\ \ x_1\ \ es\ \ A_1^k\ \wedge\ x_2\ \ es\ \ A_2^k\ \wedge \ldots \wedge\ x_m\ \ es\ \ A_m^k\ \ ENTONCES\ \ y = f^k(x_1, x_2, \ldots, x_m)$$

El antecedente es similar al de una regla del modelo Mamdani. El consecuente especifica directamente el valor numérico de salida en función de los valores numéricos de entrada. Esto hace que en este modelo la desfuzzificación quede muy simplificada mejorando la eficiencia del sistema de lógica difusa. Cada valor numérico de salida y es la media ponderada de sus apariciones en las reglas. Ésta se expresa de la siguiente manera:

$$w_k = \min\left(\mu_{A_1^k}(x_1)\ ,\ \mu_{A_2^k}(x_2)\ ,\ \ldots\ ,\ \mu_{A_m^k}(x_m)\right)$$

$$y'_k = f^k(x_1, x_2, \ldots, x_m)$$

$$y = \frac{\sum_{i=1}^{q} w_i y'_i}{\sum_{i=1}^{q} w_i}$$

El modelo Mamdani está especialmente indicado cuando el número de variables es reducido y las reglas son proporcionadas por personas expertas. En cambio el modelo Sugeno está dirigido a problemas con una elevada cantidad de variables y en los que es necesario emplear técnicas de creación automática de reglas para la formación del sistema de lógica difusa.

Una posibilidad para aproximar la función F es emplear un sistema de lógica difusa. En este caso los valores numéricos de entrada se corresponden con la secuencia de estados $\{s_n\}$ y el único valor numérico de salida con la estimación del estado siguiente $\hat{s}_{n+1}$. Para calcular el error del sistema se tiene en cuenta el valor real del estado siguiente s_{n+1}.

La predicción de sistemas caóticos deterministas mediante sistemas de lógica difusa es compatible con el modelo Mamdani [213] aunque para este tipo de labores se recomienda el modelo Sugeno siempre y cuando el ruido en los datos no sea muy elevado [214]. Por ello consideramos conveniente centrarnos en el modelo Sugeno y en concreto en el de orden cero. En éste las funciones empleadas en el consecuente de las reglas son polinomios de grado cero. Con esta restricción se facilita la creación y gestión automática de las reglas difusas.

Las cualidades de un sistema de lógica difusa están marcadas principalmente por las variables y términos lingüísticos con los que opere. Si son muy numerosas la

cantidad de reglas necesarias para contemplar todas las combinaciones posibles de variables y términos lingüísticos se dispara con la consecuente pérdida de eficiencia ($min(t) \wedge min(C)$). En cambio si son escasas se corre el riesgo de perder información y no proporcionar un resultado satisfactorio ($min(e)$). Por ello en esta situación es conveniente considerar sólo los últimos m estados de la secuencia de estados $\{s_n\}$. La Fig. 32 muestra el esquema del sistema de lógica difusa con esta característica.

En este sistema de lógica difusa cada regla asocia a cada patrón de estados una estimación del estado siguiente. De este modo una regla difusa R^k en la que γ^k es el coeficiente del polinomio de grado cero de su consecuente se expresa de la siguiente manera:

$$R^k: \; SI \; s_{n-m+1} \; es \; A_1^k \wedge ... \wedge \; s_n \; es \; A_m^k \; \; ENTONCES \; \hat{s}_{n+1} = \gamma^k$$

El número total υ de reglas difusas, contemplando cada una un patrón de estados distinto y siendo l el número de términos lingüísticos, cumple la siguiente expresión:

$$\upsilon = l^m$$

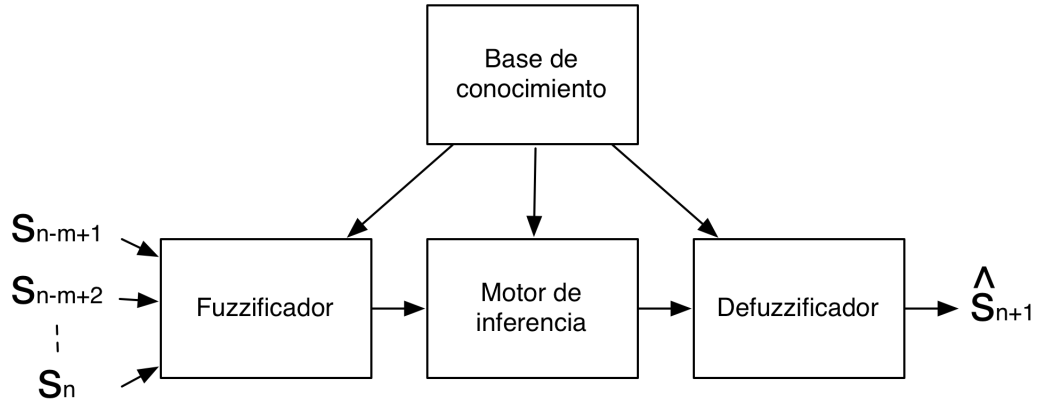

Fig. 32. Sistema de lógica difusa

Estas reglas se crean automáticamente siguiendo un esquema inductivo (patrón-estimación-regla) cuando aparecen patrones no contemplados en la base de conocimiento hasta completar las υ reglas. En cualquier caso en cada ejecución del sistema de lógica difusa se ajustan los coeficientes de los polinomios de los consecuentes de las reglas registradas [215] para poder proporcionar en todo momento una estimación del estado siguiente con el mínimo error posible.

Los coeficientes γ^k de los polinomios de los consecuentes de las reglas difusas forman la matriz de coeficientes $\boldsymbol{\Gamma} \in \mathcal{M}_{1\times\upsilon}(\mathbb{R})$. Ésta es una matriz densa de υ elementos y su expresión es la siguiente:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \gamma^1 & \gamma^2 & \dots & \gamma^\upsilon \end{bmatrix}$$

Este sistema de lógica difusa tiene ciertas ventajas destacables. La creación y ajuste de las reglas difusas es sencilla. En su base de conocimiento pueden coexistir reglas difusas con distinto valor de m, más generales (m menor) y más concretas (m mayor). Por ello la sensibilidad a pequeños patrones de la secuencia de estados es mayor que la de otros métodos.

Sin embargo, este sistema también presenta algunos inconvenientes que deben tenerse en cuenta. Para no perder precisión es adecuado que l no sea muy pequeño. Además para disminuir la ambigüedad es útil que m sea grande. Todo esto repercute en un número muy elevado de reglas. En este supuesto la eficiencia se ve muy comprometida tanto por los recursos empleados como por el tiempo necesario para verificar todas las reglas y aplicar las que se cumplan.

Otro problema es el comportamiento del sistema de lógica difusa cuando aparecen nuevos patrones no contemplados en la base de conocimiento lo cual suele ser frecuente por la elevada variabilidad del estado del enlace inalámbrico y de su

entorno. En estos casos la estimación del estado siguiente proporcionada no es fiable y debe descartarse. Esto es complicado porque el sistema de lógica difusa no realiza esta tarea y tampoco advierte de esta situación. Esperar a que ocurran todos los patrones posibles y se creen sus correspondientes reglas antes de admitir las estimaciones del estado siguiente no es viable porque no hay garantía de que el tiempo necesario para ello sea finito. Otra estrategia para mitigar este problema es combinar reglas generales con reglas concretas para comprender más patrones posibles. Sin embargo, la ambigüedad añadida por las reglas generales puede llegar a ser contraproducente.

3.5.4 Discusión

Los métodos estudiados (redes neuronales artificiales, algoritmos genéticos y sistemas de lógica difusa) se encuentran en una situación similar ante el problema que se pretende solucionar: la predicción del estado del enlace inalámbrico para la detección prematura de interrupciones de video-streaming.

En la práctica estos métodos no pueden aproximar la versión original de la función F que asocia a la secuencia completa de n estados una estimación del estado siguiente. Para superar este obstáculo es necesaria una versión simplificada de F que sí puedan aproximar y que en este caso asocia a la ventana deslizante formada por los últimos m estados una estimación del estado siguiente. La expresión de esta nueva versión es la siguiente:

$$\hat{s}_{n+1} = F\left(s_{n-m+1}, s_{n-m+2}, \ldots, s_n\right)$$

La elección del valor de m no es trivial. Si éste es demasiado bajo se pueden dar situaciones ambiguas en las que para una misma ventana de estados sean

válidas estimaciones del estado siguiente muy dispares. En cambio si éste es demasiado elevado la complejidad de la función, y por ende la del método, aumenta. La disminución de m suele mejorar la eficiencia ($min(t) \wedge min(C)$) a costa de la eficacia ($min(e)$) y viceversa. Esta relación no es estable y el valor óptimo de m varía, no siendo siempre el máximo posible que cumpla los requisitos de eficiencia como se puede presuponer. En base a esto para un método cuasi-óptimo proponemos:

1. *Filtrar estados*. Muchas veces la aparición de determinados estados hace irrelevantes a algunos de sus vecinos. Estos últimos terminan aportando poca información y son descartables. Con esta práctica se consigue reducir parcialmente la complejidad que produce un valor de m elevado. Para realizarla es necesario incluir un filtro adaptativo muy complejo con criterios desconocidos a priori que indiquen qué estados al descartarse no alteran la predicción; no es posible garantizar la estabilidad de este filtro. Combinarlo con los métodos estudiados es poco viable y puede desestabilizar enormemente el proceso global de predicción.

2. *Resumir la secuencia de estados*. Generalmente es posible analizar un segmento de la secuencia de estados y determinar sus características básicas. Éstas forman un resumen r_n del segmento. La correspondencia entre segmento y resumen es determinada por una función R. Incorporar un resumen de este tipo en la función F permite contemplar indirectamente una cantidad de estados adicional a modo de contexto sin necesidad de aumentar el valor de m para ello. Sin embargo, el establecimiento de la función R no es trivial debido a que el resumen debe servir para minimizar el error cometido por F. Con los métodos estudiados una función R efectiva debe ser ajustada pudiendo llegar a

superar en complejidad a la función F lo cual no es deseable. La combinación de estas funciones se expresa de la siguiente manera:

$$r_n = R\left(s_{n-q+1}, s_{n-q+2}, ..., s_{n-m}\right) \ , \ q > m$$
$$\hat{s}_{n+1} = F\left(r_n, s_{n-m+1}, s_{n-m+2}, ..., s_n\right)$$

En el momento de ejecutar los métodos estudiados es la necesidad de una muestra de entrenamiento adecuada el mayor obstáculo para que operen con eficacia. El tiempo necesario para obtener esta muestra, como ya se ha comentado, puede ser elevado. La posibilidad de realizar un pre-entrenamiento en cualquiera de estos métodos no es viable porque las condiciones encontradas en el momento de ejecución suelen ser muy dispares. Por ello siempre es necesario esperar hasta lograr una adaptación suficiente. Este problema no tiene una solución sencilla pero para un método cuasi-óptimo proponemos ciertas propiedades que pueden mitigarlo:

1. *Sesgos básicos iniciales.* Ante la ausencia de experiencia en un tipo de situación es conveniente disponer previamente de algunas indicaciones básicas y generales sobre cómo ésta va a evolucionar. Estas indicaciones a modo de sesgos en la estimación del estado siguiente más que orientadas a una mayor exactitud deben estarlo a la minimización de los efectos negativos que produce una estimación incorrecta. Contemplar estos sesgos con los métodos estudiados supone combinarlos con otros métodos para realizar estimaciones del estado siguiente sesgadas y detectar las situaciones en las que considerarlas al no disponer de la suficiente experiencia. Esta ampliación no es sencilla y aumenta la complejidad del proceso general lo que puede llegar a comprometer su eficiencia.

2. *Conocimiento acumulativo.* Los métodos estudiados deben adaptarse continuamente. Esto hace que en su entrenamiento ininterrumpido los

estados recientes tengan preferencia sobre los antiguos pudiendo llegar a perderse información. Por ello el conocimiento en los citados métodos tiende acusadamente a la autodestrucción ante situaciones nuevas. Evitando esta tendencia y haciendo que el conocimiento sea acumulativo y no destructivo se evita además de la pérdida de información la necesidad de readaptarse ante situaciones ya pasadas. Proporcionar esta capacidad a los métodos estudiados implica replicarlos para cada situación diferenciada lo cual además de complicado, al tener que delimitar cada situación, es muy ineficiente por la enorme cantidad de situaciones posibles.

Una vez identificadas las carencias de los métodos analizados y propuestas las características que debe incorporar un método cuasi-óptimo derivado de ellos se puede optar por su creación o por reducir el planteamiento del problema en la medida de lo posible.

Un ejemplo reciente de reducción del planteamiento del problema es el de [216] que considera sólo el RSSI como medida para caracterizar el enlace inalámbrico. Sus autores suponen que el RSSI tiene un comportamiento de reversión a la media con saltos discontínuos. Mediante una versión modificada del proceso de tiempo continuo de Ornstein-Uhlenbeck, análogo al proceso de tiempo discreto AR(1), y un análisis probabilístico de los saltos discontínuos predicen en tiempo real la degradación o fallo del enlace inalámbrico.

Estos autores analizan el caso de un computador personal portátil en movimiento dentro de una oficina con un punto de acceso y en exterior con dos puntos de acceso separados 50 metros. Con una frecuencia de muestreo de 10 Hz tienen en cuenta sólo una ventana temporal de 3 segundos y predicen a intervalos de 0.5 segundos.

Consideramos que la predicción de interrupciones de video-streaming, con dispositivos móviles que se mueven libremente tanto en interiores como en exteriores y en presencia de uno o más puntos de acceso así como de otros dispositivos móviles, tiene una complejidad mayor que la analizada por estos autores y no se puede realizar sólo con secuencias de valores de RSSI. El resto de variables observables también debe ser tenido en cuenta. Además consideramos que aunque es posible predecir estas variables individualmente, sin contemplar relaciones entre ellas, es más conveniente en casos como éste un análisis multivariable [217].

3.6 Sistema emocional

Tradicionalmente ha existido cierto antagonismo irreductible entre la razón y la emoción. Este dualismo imperante en el pensamiento occidental ha devaluado lo considerado como emocional frente a lo racional [218]. Un caso extremo se observa en Descartes quien afirma que "la razón es la única cosa que nos hace hombres y nos distingue de los animales" [219]. Sin embargo, dos siglos después Nietzsche considera que "Descartes, padre del racionalismo, reconoció autoridad únicamente a la razón: pero ésta no es más que un instrumento, y Descartes era superficial" [220], es decir, Descartes no hizo un análisis lo suficientemente profundo: no tuvo en cuenta con detalle las fuentes condicionantes básicas del comportamiento humano. Todo esto hace recordar a otro autor contemporáneo a Descartes, Pascal, quien sostuvo intuitivamente que "el corazón tiene razones, que la razón no conoce" [221], y que por lo tanto los instintos y las emociones también deben tenerse en cuenta para llegar al máximo potencial intelectual del ser humano.

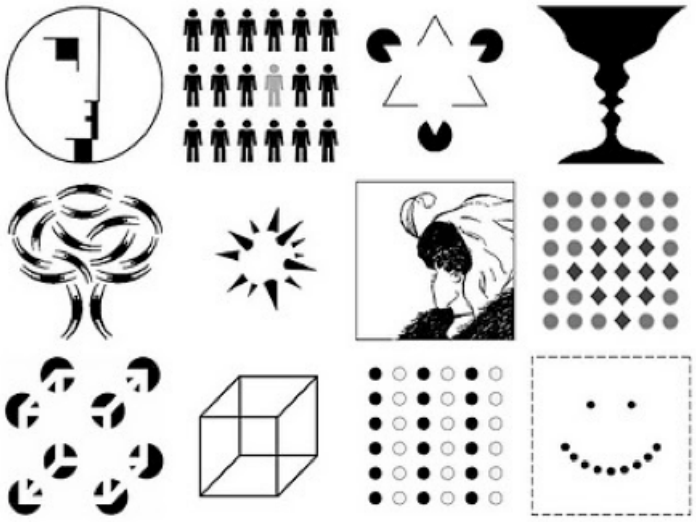

Fig. 33. Imágenes de interpretación múltiple

Otro aspecto controvertido es la relación razón-percepción. Descartes considera en [219] que "ni la imaginación ni los sentidos pueden asegurarnos nunca cosa alguna, como no intervenga el entendimiento" y que en definitiva "los sentidos nos engañan". Este pensamiento fue contradicho posteriormente por Nietzsche en [222] cuando dijo que "los sentidos no mienten de ninguna manera" y "la "razón" es la causa de que nosotros falseemos el testimonio de los sentidos". Esta última afirmación se basa en que el proceso de percepción se fundamenta en la recepción de estímulos objetivos y su posterior interpretación subjetiva. De esta manera el sujeto es capaz de crear un mapa mental de su entorno en el que además de la información proporcionada por los estímulos recibidos se incorpora metainformación derivada del propio mapa y del conocimiento adquirido con anterioridad. Esta metainformación condiciona la interpretación del entorno, la atención prestada sobre el mismo y la manera en la que es memorizado para poder interactuar con él. Por ejemplo, en psicología de la Gestalt [223] se estudia este fenómeno con imágenes. La Fig. 33 [225] es una composición de imágenes populares

en la psicología de la Gestalt que constatan el carácter subjetivo de la percepción. Cada una de estas imágenes tiene más de una interpretación posible.

El antagonismo entre razón y emoción no está justificado y como se argumenta en [224] las emociones pueden considerarse como un tipo de percepción con gran peso en las creencias racionales, los deseos y la toma de decisiones.

3.6.1 Estado del arte

En un proceso de toma de decisiones puramente racional no se tienen en cuenta las emociones y éstas suelen considerarse un estorbo. Este proceso queda reducido a la elección de la mejor alternativa posible según la valoración de cada una de ellas en términos de pros y contras, ganancias y pérdidas o ventajas y desventajas. Esta visión tan estoica no se corresponde completamente con la manera habitual de tomar decisiones de un ser humano. Según la neurofisiología y la neuropsicología existe una participación apreciable de las emociones [226] en los mecanismos cerebrales implicados en la toma de decisiones [227].

Los considerados aspectos racionales de la toma de decisiones en un ser humano se sirven de múltiples procesos cognitivos y funciones ejecutivas [228]. Los procesos cognitivos son procedimientos mentales básicos implicados en la incorporación y el uso de conocimiento. La percepción, la comprensión, la memorización, el razonamiento lógico o el aprendizaje son procesos cognitivos básicos en los que se sustentan prácticas más complejas como las funciones ejecutivas. Éstas son habilidades que permiten decidir a partir de la percepción del entorno qué acciones realizar para la consecución de un objetivo concreto. Ejemplos de estas habilidades son la atención en la tarea realizada inhibiendo información no relevante, la memorización selectiva de la experiencia obtenida, el diálogo interior

dedicado a la orientación y control de la conducta, y la instauración de motivaciones que justifiquen el esfuerzo realizado para alcanzar el objetivo final.

Actualmente se considera que las emociones complementan y potencian los procesos cognitivos implicados en la toma de decisiones. La valoración de las emociones ha sido progresiva pasando de no ser tomadas en cuenta a reconocer su importancia en el pensamiento humano [229]. Experimentalmente se ha constatado la relación entre capacidad emocional y toma de decisiones. Si la capacidad emocional se ve reducida por lesiones en el córtex prefrontal ventromedial o por el abuso de drogas duras la toma de decisiones se ve afectada [230] quedando ésta deteriorada en cualquier caso [231].

Las emociones tienen gran influencia en el comportamiento del sujeto que las experimenta llegando a condicionar sus objetivos [232] y su experiencia social en su grupo y en su entorno en general [233]. En una toma de decisiones realizada por una persona son consideradas las emociones relacionadas con situaciones similares, con el contexto y con la valoración de las consecuencias predichas de cada alternativa. Las emociones sirven de guía eficaz de la toma de decisiones simplificando el proceso y aumentando su velocidad. También atenúan los conflictos producto de alternativas con valor similar. Un modelo neurocognitivo ampliamente aceptado que incluye estos vínculos entre emociones y toma de decisiones es el basado en la hipótesis del marcador somático de Damasio [234] y su verificación mediante la *Iowa Gambling Task* (IGT) [235].

Según Damasio cuando existen varias alternativas de actuación el cerebro crea fugazmente representaciones de las situaciones futuras que pueden originar. Cada una de estas representaciones tiene asociado un marcador somático. Un marcador somático se define como “un cambio corporal que refleja un estado

emocional, ya sea positivo o negativo". De esta forma se relaciona una situación con un estado emocional a través de un marcador somático. Esta relación permite valorar emocionalmente las alternativas, descartar las más desagradables o negativas y seleccionar finalmente la más agradable o positiva. Este proceso es mucho más rápido que uno equivalente puramente racional debido a que la valoración emocional de las alternativas se realiza inconscientemente. Con ello es posible obtener resultados altamente satisfactorios sin necesidad de estrategias explícitas [236]. La eficacia de esta toma de decisiones suele ser elevada en la mayoría de situaciones cotidianas al estar apoyada en toda la experiencia personal adquirida y no sólo en un procedimiento racional lógico y mecánico.

Bechara [237] distingue en el cerebro humano dos sistemas que interactúan entre sí para la toma de decisiones: impulsivo y reflexivo. El sistema impulsivo, situado en la amígdala, se encarga de las decisiones que deben tomarse inmediatamente sin demora. Opera en términos de dolor-placer exclusivamente. Su ejecución es inconsciente. El sistema reflexivo, situado en el córtex prefrontal ventromedial, se encarga del resto de decisiones que pueden esperar cierto intervalo de tiempo. Este sistema incorpora aspectos como las motivaciones, la experiencia vital o la inhibición de conductas. Gran parte de su ejecución es consciente y define la voluntad del sujeto. El equilibrio entre estos dos sistemas condiciona el comportamiento final del individuo situándose entre la impulsividad y la reflexividad.

El aumento progresivo de la comprensión de las emociones [238] ha hecho posible su uso en inteligencia artificial para resolver problemas en los que los métodos clásicos no proporcionan buenos resultados. Dotar a agentes software de emociones [239] y de la capacidad de servirse de ellas para sociabilizarse de forma efectiva [240] se muestra como una posibilidad positiva. Para ello se han propuesto

diferentes arquitecturas emocionales que han ido evolucionando con el tiempo. En [241] se enumeran las más representativas clasificándolas en cognitivas y no cognitivas. Como arquitectura no cognitiva se encuentra la basada en el modelo de Tomkins [242], y como cognitivas las basadas en los modelos de Simon [243], Sloman [244] y Frijda [245].

En una arquitectura emocional no cognitiva se considera una emoción como una reacción automática e involuntaria tipo estímulo-respuesta. En el modelo de Tomkins la emoción concreta que se experimenta depende de la densidad de impulsos nerviosos la cual está relacionada con la intensidad del estímulo. A mayor densidad se dan emociones más negativas y viceversa. De este modo ante una densidad constante se pueden dar emociones como la angustia o el enfado, ante una densidad creciente el interés, el miedo o el susto y ante una densidad decreciente la diversión o la alegría.

En una arquitectura emocional cognitiva las emociones se entienden como procesos evaluadores del entorno que forman un sistema de vigilancia en permanente funcionamiento destinado a la detección de situaciones de atención urgente. Se siguen considerando reacciones automáticas e involuntarias pero en este caso originadas por la valoración cognitiva de la situación presente [246]. Según el modelo de Simon las emociones pueden interrumpir los procesos en curso y forzar al sistema a que centre todos sus recursos en la nueva situación detectada. El modelo de Sloman flexibiliza este mecanismo de interrupción añadiendo un filtro de atención variable dependiente del contexto. Esto hace que cuando se manifieste una emoción ésta sólo acapare una parte de los recursos disponibles dejando el resto para los procesos en curso. El modelo de Frijda añade un mecanismo de control del filtro de atención en base a la mejora de la gestión de recursos y a los objetivos actuales en el momento presente.

Las arquitecturas emocionales no cognitivas han tenido un éxito discreto debido a sus limitaciones y a que sólo contemplan un número pequeño de emociones modeladas como simples respuestas automáticas y predefinidas. Las arquitecturas emocionales cognitivas han conseguido una mayor difusión destacando la *Tractable Appraisal-Based Architecture for Situated Cognizers* (TABASCO) [247] la cual está basada en el modelo de Frijda. Sin embargo, estas arquitecturas siguen siendo demasiado limitadas. Al modelar emociones como estados predefinidos se restringe el universo emocional. Además sólo se tienen en cuenta aquellas capaces de producir perturbaciones o interrupciones en el sistema. Otro problema de estas arquitecturas es que no están pensadas para integrar procesos de aprendizaje o de toma de decisiones que utilicen emociones.

En [241] se dan algunas recomendaciones para el desarrollo de arquitecturas emocionales más flexibles y potentes considerando los avances que se han dado en la neurofisiología y neuropsicología actuales. La primera de ellas es superar el dualismo razón-emoción y centrar los esfuerzos en estudiar las emociones desde un punto de vista funcional evitando enfoques mecanicistas e interpretativos. La segunda es no considerar las emociones como perturbaciones sino como mecanismos de apoyo a la razón tal y como se plantea en el modelo basado en la hipótesis del marcador somático de Damasio. En la arquitectura emocional sugerida las emociones no son estados predefinidos sino procesos funcionales. Son consideradas mecanismos de supervivencia que mejoran el comportamiento del sistema y que forman parte de los procesos de toma de deciciones.

3.6.2 Modelo emocional

Existen modelos emocionales en los que se consideran las emociones como fenómenos aislados [248] lo cual reduce su efectividad. Solventar esta limitación requiere un modelo con un enfoque holístico: un modelo de la combinación intuición-pensamiento. En la actualidad no se dispone del conocimiento suficiente para su desarrollo con detalle y exactitud pero es posible proponer modelos aproximados según lo que se sabe del tema a día de hoy.

Hay multitud de estudios de neurociencia en sus diferentes áreas [249]. Gran parte de ellos muestran los resultados obtenidos a partir de la observación de individuos o animales con sus funciones alteradas por lesiones, estimulación electromagnética del cerebro, técnicas psicológicas, psicofármacos o incluso implantado *microchips* [250]. Las técnicas más utilizadas en este caso son la imagen por resonancia magnética funcional (fMRI) [251], la electroencefalografía (EEG) [252], la magnetoencefalografía (MEG) [253], la topografía de estado estable (SST) [254], el seguimiento de ojos (*Eye Tracking*) [255] o diferentes medidas fisiológicas como la frecuencia cardíaca o la actividad electrodérmica (EDA) [256]. Los modelos matemáticos no son la herramienta de investigación principal en este campo salvo en neurociencia computacional [257] donde se utilizan para simular el comportamiento de las neuronas de forma realista. Éstos suelen ser modelos dinámicos no lineales [258] y de cierta complejidad matemática [259] lo cual no es deseable para su aplicación en inteligencia artificial. De momento no llegan a integrar eficazmente procesos cognitivos aunque distintos proyectos en los que se pretende simular el cerebro a nivel molecular como el *Blue Brain Project* [260] (cerebro de mamíferos) o el *Human Brain Project* [261] (cerebro humano) están acercando esta posibilidad. Mientras tanto el vacío de conocimiento existente entre

el nivel neuronal y el nivel cognitivo es el mayor desafío actual para el desarrollo de un modelo de la combinación intuición-pensamiento. El modelo descriptivo propuesto en este trabajo sobre ello aunque hipotético es coherente con el conocimiento disponible actualmente en neurociencia cognitiva [262], haciendo especial énfasis en los procesos cognitivos vinculados a la toma de decisiones. Éstos son estudiados en profundidad en otras áreas como la neuroeconomía [263] o el neuromarketing [264] tratando sobre todo el equilibrio entre los sistemas de recompensa cerebral y de aversión a la pérdida [265].

El estudio de los procesos cognitivos desde una perspectiva neurocientífica se hace principalmente en base a la activación secuencial de las diferentes regiones del cerebro [266]. Estas regiones están delimitadas, muy especializadas en un tipo de información y conectadas unas con otras para colaborar entre ellas. Las secuencias que describen son modelables y para ello es posible emplear distintos métodos probabilísticos [267]. Por ejemplo, con redes bayesiadas [268] se ha modelado la interacción de distintos niveles del cortex visual dedicados al reconocimiento de imágenes [269]. En este caso éstos guardan una relación jerárquica en la que los niveles inferiores detectan patrones de una imagen más concretos y particulares, principalmente los puntos de mayor curvatura del contorno [270], y dependiendo de cómo se activen éstos los niveles superiores detectan patrones más abstractos y generales. Según Hawkins [271] esta relación jerárquica es común en el resto de regiones cerebrales centrándose las inferiores en el reconocimiento y las superiores en la predicción del futuro cercano. De este modo entendemos la mente como el resultado de la ejecución en paralelo de una cantidad colosal de procesos tipo reconocimiento-predicción los cuales forman una sociedad como la propuesta por Minsky [272]. Estos procesos se repiten y encadenan uno detrás de otro orquestándose y manteniendo una coherencia sincronizándose. Para modelarlos hay

que guardar especial cuidado con este aspecto tal y como también se hace en los modelos a nivel neuronal [273].

Según Kahneman [274] la mente sigue dos vías independientes capaces de interactuar: la intuición y el pensamiento. En este trabajo no hacemos esta separación tan marcada, aunque si una sucesión de procesos encadenados reconocimiento-predicción es corta, concluye rápidamente y es imperceptible como tal por el sujeto que la experimenta la identificamos como una intuición. En cambio si la sucesión es prolongada y perceptible la consideramos un pensamiento. De este modo integramos la intuición y el pensamiento en una sola vía.

Antes de comenzar la especificación del modelo planteado es necesario concretar la definición de emoción. Esto es complicado porque no existe una definición consensuada y prácticamente cada autor tiene la suya. Adoptamos la de Wukmir [275] por su aceptación, simplicidad y concisión:

"Respuesta inmediata del organismo que le informa del grado de favorabilidad de un estímulo o situación".

Como complemento también consideramos la definición de emoción de Oatley [276]:

"Experiencia afectiva en cierta medida agradable o desagradable, que supone una cualidad fenomenológica característica y que compromete tres sistemas de respuesta: cognitivo-subjetivo, conductual-expresivo y fisiológico-adaptativo".

Combinando estas definiciones una emoción (Fig. 34) es una respuesta inmediata e inconsciente constituida por una sensación subjetiva, una conducta expresiva y un cambio fisiológico que en conjunto producen una experiencia afectiva que refleja una medida de supervivencia, agradable ante situaciones

favorables y desagradable ante situaciones hostiles. Conductas expresivas asociadas a emociones son algunos gestos, posturas o verbalizaciones, y cambios fisiológicos característicos la alteración del ritmo cardiaco, la alteración del ritmo respiratorio, sudoración o lagrimeo entre otros.

Las emociones tienen tres funciones principales [277]:

1. *Adaptativa*. Los cambios fisiológicos producto de las emociones adecúan el estado físico del sujeto a la situación que esté viviendo. Por ejemplo, minimizan el uso de energía en situaciones de calma y aumentan su disponibilidad en situaciones de amenaza.

2. *Social*. La expresión de emociones conscientemente, mediante verbalizaciones, o inconscientemente, mediante gestos, amplían la comunicación interpersonal y mejoran la convivencia [278].

3. *Motivacional*. La conducta de un individuo está marcada por la búsqueda de emociones positivas en un futuro lo cual teóricamente equivale al aumento de su probabilidad de supervivencia.

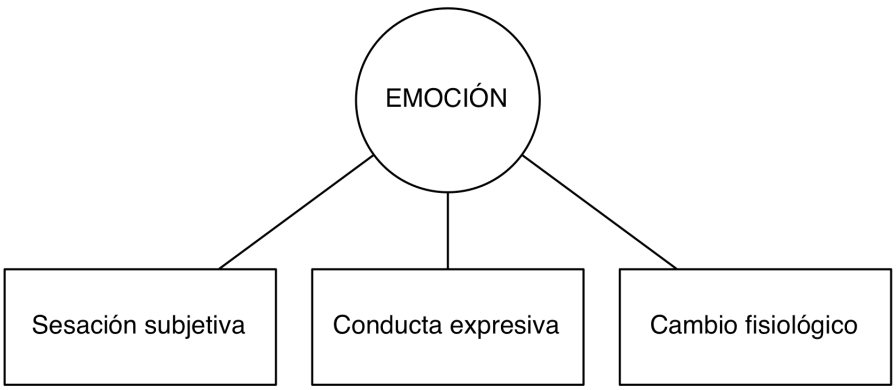

Fig. 34. Componentes de una emoción

La delimitación, caracterización y diferenciación de las emociones es un tema controvertido [279]. Por ello algunos autores han evitado clasificar las emociones en categorías predefinidas y han optado por representarlas cuantitativamente en base a sus características esenciales como puntos de un espacio matemático multidimensional. Uno de los primeros en hacer esto fue Wundt [280] quien propuso un espacio tridimensional: agrado-desagrado, tensión-relajación, excitación-calma. Actualmente el espacio emocional más aceptado es el espacio bidimensional: agrado-desagrado, intensidad [281] pero consideramos que éste es demasiado simple y un punto en él puede corresponderse con varias emociones diferentes. Para solventar esta falta de precisión hay que añadir información cualitativa lo que no es deseable. Esto limita las ventajas que proporciona una caracterización completamente cuantitativa de las emociones. Por ello proponemos un espacio métrico nuevo con algunas operaciones y funciones asociadas tras suponer la existencia de las siguientes relaciones:

1. *Relación consumo de energía - probabilidad de supervivencia.* La labor del cerebro es maximizar la probabilidad de supervivencia del individuo con un consumo mínimo de energía. Un incremento de este consumo sólo está justificado en situaciones de baja o alta probabilidad de supervivencia si con éste se consigue aumentarla o mantenerla respectivamente. Por lo tanto a mayor consumo de energía más extrema es la probabilidad de supervivencia y viceversa. A este nivel esta relación es demasiado ambigua por lo que es necesario un mayor detalle.
2. *Relación activación de regiones cerebrales - estado emocional.* La mayor parte de la energía empleada se dedica a la generación de impulsos nerviosos asociados a la activación de las diferentes regiones cerebrales. Estas activaciones son diferentes según las causen situaciones favorables o

desfavorables por lo que conociéndolas es posible distinguirlas como activaciones positivas o negativas respectivamente. De este modo cada activación tiene un signo y una intensidad caracterizada por la cantidad de impulsos nerviosos que tiene asociada. A partir del conjunto de activaciones es posible determinar la probabilidad de supervivencia pero no a la inversa. Para ello es necesario expresar la probabilidad de supervivencia con más detalle como un estado emocional. Con estas precisiones es factible reescribir la relación entre consumo de energía y probabilidad de supervivencia, como se ilustra en la Fig. 35, reduciendo su ambigüedad.

Consideramos que la relación entre activación de regiones cerebrales y estado emocional es una correspondencia biunívoca. Por ello, un estado emocional queda representado sin ambigüedad por su correspondiente especificación de activaciones. Para definir un estado emocional utilizamos el espacio vectorial normado η-dimensional E^{η} sobre $\mathbb{R}$ siendo η el número de regiones cerebrales. Este espacio es el conjunto de todas las tuplas ordenadas $\left(x_1,x_2,...,x_{\eta}\right)$ en las que cada $x_i \in \mathbb{Z}$ tiene por signo $\mathrm{sgn}\left(x_i\right)$ el de la activación de la región cerebral i y por módulo $\left|x_i\right|$ la cantidad de impulsos nerviosos de esa activación. Las operaciones y funciones asociadas a este espacio son la ley de composición interna suma $+$, la ley de composición externa producto por un peso escalar $\cdot$, la ley de composición externa producto por un peso vectorial $*$, la operación externa cociente $\div$, la operación norma $\|\cdot\|$, la función distancia d, la función parecido p y la función reducción r. Con ello este espacio vectorial normado queda representado completamente por la tupla:

$$\left(E^{\eta},\ +,\cdot,\ *,\ \div,\ \|\cdot\|,\ d,\ p,\ r\right)$$

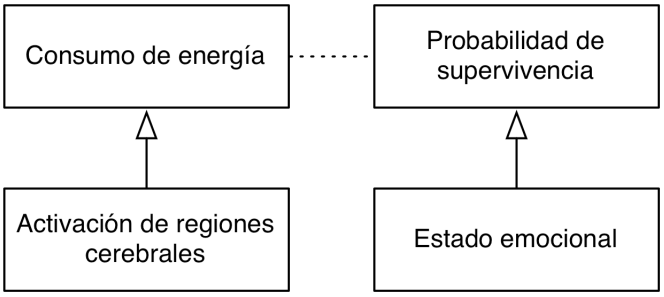

Fig. 35. Relación consumo energía - probabilidad de supervivencia

La ley de composición interna suma + se define como:

$$+ : E^{\eta} \times E^{\eta} \rightarrow E^{\eta}$$
$$(\mathbf{x},\mathbf{y}) \rightarrow \mathbf{z} = \mathbf{x} + \mathbf{y}$$
$$z_i = x_i + y_i$$

Esta operación combina dos estados emocionales. En ella queda reflejado que si las activaciones de una misma región cerebral son del mismo signo sus impulsos nerviosos se suman. En caso contrario se contrarrestan.

La ley de composición externa producto por un peso escalar · se define como:

$$\cdot : \mathbb{R} \times E^{\eta} \rightarrow E^{\eta}$$
$$(w,\mathbf{x}) \rightarrow \mathbf{y} = w \cdot \mathbf{x}$$
$$y_i = wx_i$$

Esta operación pondera un estado emocional con un peso escalar real.

La ley de composición externa producto por un peso vectorial $*$ se define como:

$$* : \mathbb{R}^{\eta} \times E^{\eta} \rightarrow E^{\eta}$$
$$(\mathbf{w},\mathbf{x}) \rightarrow \mathbf{y} = \mathbf{w} \cdot \mathbf{x}$$
$$y_i = w_i x_i$$

Esta operación pondera un estado emocional con un peso vectorial real η-dimensional. Permite dar un peso diferente a cada activación.

La operación externa cociente $\div$ se define como:

$$\begin{aligned} \div : E^{\eta} \times E^{\eta} &\rightarrow \mathbb{R}^{\eta} \\ (\mathbf{x},\mathbf{y}) &\rightarrow \mathbf{w} = \mathbf{x} \div \mathbf{y} \end{aligned}$$

$$w_i = \frac{x_i}{y_i}$$

Esta operación proporciona la razón existente entre dos estados emocionales η-dimensionales expresada en forma de peso vectorial real η-dimensional.

La operación norma $\|\cdot\|$ se define como:

$$\begin{aligned} \|\cdot\| : E^{\eta} &\rightarrow \mathbb{R} \\ \mathbf{x} &\rightarrow \sum_{i=1}^{\eta} |x_i| \end{aligned}$$

Esta operación proporciona el número total de impulsos nerviosos asociado a un estado emocional. Esta norma es del tipo norma L_1 de la cual se deriva la distancia Manhattan [282].

La función distancia d entre dos puntos se define como:

$$\begin{aligned} d : E^{\eta} \times E^{\eta} &\rightarrow \mathbb{R} \\ (\mathbf{x},\mathbf{y}) &\rightarrow \|\mathbf{x} - \mathbf{y}\| = \sum_{i=1}^{\eta} |x_i - y_i| \end{aligned}$$

Esta función distancia equivale a la suma de las distancias entre activaciones considerando tanto su signo como su número de impulsos nerviosos. Se corresponde con la distancia Manhattan quedando definida a partir de la operación norma.

La función parecido p entre dos puntos se define como:

$$p : E^{\eta} \times E^{\eta} \rightarrow [0,1]$$

$$(\mathbf{x},\mathbf{y}) \rightarrow \frac{\sum_{i=1}^{\eta} \begin{cases} \min(|x_i|,|y_i|) & , \; si \;\; x_i y_i > 0 \\ 0 & , \; si \;\; x_i y_i \leq 0 \end{cases}}{\sum_{i=1}^{\eta} \max(|x_i|,|y_i|)}$$

Esta función parecido expresa la similitud entre dos estados emocionales con un valor entre 0 y 1 indicando ninguna o total similitud respectivamente. Para su definición generalizamos la función de Ruzicka, perteneciente a la familia de las intersecciones [282], contemplando en este caso números de distinto signo.

La función reducción r se define como:

$$r : E^{\eta} \rightarrow \mathbb{Z}$$

$$\mathbf{x} \rightarrow \sum_{i=1}^{\eta} x_i$$

Esta función equivale a la suma de todas las activaciones de un estado emocional reduciéndolo a una sola activación. Esta activación resultante define el signo del estado emocional y su intensidad.

$$\mathbf{x} \in E^{\eta}$$
$$\operatorname{sgn}(\mathbf{x}) = \operatorname{sgn}(r(\mathbf{x}))$$
$$|\mathbf{x}| = |r(\mathbf{x})|$$

La relación cognición-emoción es otro aspecto problemático a estudiar. Las emociones tienen un origen evolutivo y en seres humanos se ha observado que su forma no depende de la raza o la cultura [283] e incluso de la voluntad de quien las experimenta. Este hecho se ha aprovechado, por ejemplo, para la detección de mentiras cuando aparecen incoherencias como cambios fisiológicos injustificados o microgestos contradictorios [284]. La relación cognición-emoción es muy estrecha

[285] existiendo en todo momento una correspondencia entre lo que se está pensando y las emociones que se experimentan. Al mentir las emociones muchas veces revelan lo que de verdad se piensa. Pero esta relación es bidireccional [286] y la cognición además de influir en la activación de las emociones se ve modulada e incluso determinada por éstas. Diversos autores han encontrado evidencias de esta relación [287] aunque sin mucho detalle por las dificultades y limitaciones que presentan las metodologías de investigación disponibles hasta el momento en este campo.

La cognición es la encargada de procesar la información proporcionada por los sentidos y por el conocimiento derivado de la experiencia. Este conocimiento es un esquema o mapa que representa la realidad del individuo. Su finalidad es permitir interpretar nuevas situaciones y actuar en consecuencia racionalmente. La memoria se encarga de almacenarlo y gestionarlo siendo la pieza fundamental y el sustento de la cognición, es decir, de procesos como la percepción, el aprendizaje, la creatividad o el razonamiento. Según la psicología [288] la memoria se organiza en varios niveles interconectados. Broadbent con su modelo de flujo de información con filtros selectivos [289] y posteriormente Atkinson y Shiffrin con su modelo de memoria de almacenamiento múltiple [290] fueron los precursores de esta idea. Ésta se basa en que la memoria está organizada en tres niveles diferenciados por su duración y funcionalidad: memoria sensorial, memoria a corto plazo y memoria a largo plazo. La memoria sensorial almacena la información suministrada por los sentidos durante un período de tiempo muy corto, del orden de décimas de segundo. Ésta es filtrada por el proceso de atención y el resultado es interpretado y asociado a elementos abstractos manejables. Estos elementos se ubican en la memoria a corto plazo. La capacidad de esta memoria es muy limitada, en torno a 7 elementos, y su duración es de unos 10 segundos si el contenido no es reforzado

por repetición. Finalmente sólo algunos elementos son retenidos permanentemente pasando a la memoria a largo plazo y pudiendo ser recuperados posteriormente. La memoria a largo plazo posee una capacidad y duración indefinida. El conocimiento que almacena se divide en procedimental y declarativo, y éste último en episódico y semántico. El conocimiento procedimental se corresponde con acciones realizables, el declarativo episódico con secuencias ordenadas temporalmente de hechos pasados y el declarativo semántico con conceptos abstractos interrelacionados según su significado. En la Fig. 36 se ilustra este modelo de memoria de tres niveles.

Este modelo aunque aceptado presenta debilidades notables al estar más influenciado por la "metáfora del computador" que por la propia biología. Hace más énfasis en aspectos estructurales que en los procesos cognitivos. Para mejorar esta cuestión Baddeley y Hitch introdujeron el concepto de memoria de trabajo [291] la cual, muy relacionada con la memoria a corto plazo, sigue guardando un elevado paralelismo con la RAM de un computador. El problema de esta metáfora reside en que la memoria de un ser vivo no trabaja de la misma manera que la de una máquina: no están destinadas a los mismos fines.

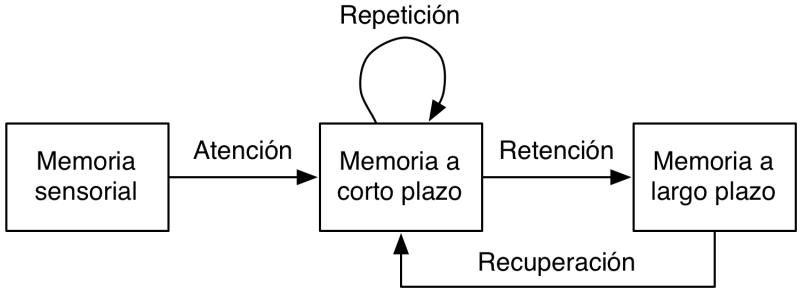

Fig. 36. Modelo de memoria de tres niveles

La memoria de un ser vivo es una herramienta de propósito específico que construye y almacena conocimiento útil en forma de eventos interrelacionados [292] para emplearlo en todo momento aumentando la probabilidad de supervivencia. La memoria de una máquina es una herramienta de propósito general que guarda información en forma de secuencia de datos para ser consultada cuando se requiera.

Estructuralmente estas memorias son muy diferentes. La memoria de un ser vivo es autoconfigurable fruto de la plasticidad neuronal. Con ello persigue un menor consumo energético para no aumentar las necesidades alimenticias y un mayor paralelismo para procesar más información. La recuperación de información la realiza asociativamente por contenido al no ser posible establecer un sistema de direccionamiento de memoria. En cambio la estructura de la memoria de una máquina es rígida lo que le proporciona una máxima capacidad de almacenamiento y un mínimo tiempo de acceso a la información. En este caso la recuperación de información sí se realiza por medio de direcciones de memoria.

Las emociones influyen en la recuperación de información de la memoria y en sus procesos cognitivos asociados [293]. Son responsables del efecto de congruencia con el estado emocional. Éste hace que sea más probable recordar estímulos con la misma tonalidad afectiva que la del estado emocional actual y entre ellos primero aquellos asociados a estados emocionales de intensidad superior.

Las emociones permiten a la memoria organizar lo percibido en forma de eventos reconocibles. Los límites de estos eventos están marcados por los momentos en los que el error de predicción cometido en los múltiples procesos cerebrales reconocimiento-predicción ha sido significativamente más elevado [294]. Este error produce una emoción negativa conocida como disonancia cognitiva [295]. La creación de estos eventos es la base del proceso cognitivo de abstracción.

Consideramos que la memoria es el elemento estructural que almacena la secuencia de estados emocionales vividos. En ésta no distinguimos niveles concretos como memoria a corto plazo o a largo plazo. En cambio cada estado emocional almacenado tiene asociado un nivel o medida de presencia que lo hace más o menos evocable. Definimos el contenido de la memoria como un elemento del conjunto M^n en el que n es el número de estados emocionales almacenados. Cada elemento de este conjunto es una tupla ordenada de pares $\left((\mathbf{x}_1,\rho_1),(\mathbf{x}_2,\rho_2),...,(\mathbf{x}_n,\rho_n)\right)$ en los que $\mathbf{x}_i \in E^\eta$ y $\rho_i \in \mathbb{R}^+ \cup \{0\}$ son el estado emocional almacenado con índice i y su indicador de presencia asociado respectivamente.

De acuerdo con el efecto de congruencia el acceso a la memoria es asociativo con respecto al estado emocional. La memoria proporciona la estimación del estado emocional siguiente $\hat{\mathbf{x}}$ según los estados emocionales evocados a partir del estado emocional $\mathbf{x}$. Este proceso queda representado por la función ψ la cual definimos como:

$$\begin{aligned} \psi : M^n \times E^\eta &\rightarrow M^{n+1} \times E^\eta \\ (\mathbf{b},\mathbf{x}) &\rightarrow (\mathbf{b}',\hat{\mathbf{x}}) = \psi(\mathbf{b},\mathbf{x}) \end{aligned}$$

Debido a la complejidad de esta función es conveniente descomponerla de la siguiente manera:

$$\begin{aligned} \psi(\mathbf{b},\mathbf{x}) &= (\chi \circ \xi \circ \varsigma)(\mathbf{b},\mathbf{x}) \\ \psi(\mathbf{b},\mathbf{x}) &= (\mathbf{b}',\hat{\mathbf{x}}) \end{aligned}$$

La función ς agrega a la memoria $\mathbf{b}$ el estado emocional $\mathbf{x}$.

$$\begin{aligned} \varsigma : M^n \times E^\eta &\rightarrow M^{n+1} \\ (\mathbf{x},\mathbf{b}) &\rightarrow \mathbf{b}' = \varsigma(\mathbf{b},\mathbf{x}) \end{aligned}$$

Esta función añade a la memoria $\mathbf{b}$ con n estados emocionales almacenados el estado emocional $\mathbf{x}$ con índice $n+1$. El indicador de presencia asignado a este nuevo estado es el mínimo posible.

$$\varsigma(\mathbf{x},\mathbf{b}) = \mathbf{b} \cup (\mathbf{x},0)$$

La función ξ modifica los indicadores de presencia de los estados emocionales almacenados en la memoria $\mathbf{b}$ en base al último estado emocional agregado.

$$\begin{aligned} \xi &: M^n \rightarrow M^n \\ \mathbf{b} &\rightarrow \mathbf{b}' = \xi(\mathbf{b}) \end{aligned}$$

El primero de ellos no cambia al depender cada indicador de presencia de su correspondiente precedente.

$$\begin{aligned} &\mathbf{b}' = \xi(\mathbf{b}) \\ &b_i = (\mathbf{x}_i, \rho_i) \\ &b'_i = \begin{cases} (\mathbf{x}_i, 0) & , \; si \;\; i = 1 \\ (\mathbf{x}_i, \rho'_i) & , \; si \;\; i > 1 \end{cases} \end{aligned}$$

Para cada estado emocional almacenado con índice i distinto del primero el nuevo indicador ρ'_i asociado se descompone en un indicador ρ_i^+ de evocación y un indicador ρ_i^- de olvido.

$$\rho'_i = \rho_i^+ + \rho_i^-$$

El indicador ρ_i^+ representa el resultado de la búsqueda asociativa del último estado emocional agregado $\mathbf{x}_n$ con respecto al estado emocional almacenado con índice i. Su valor es proporcional a la intensidad $|\mathbf{x}_i|$ del estado emocional almacenado y a su parecido con el último estado emocional agregado.

$$\rho_i^+ \propto |\mathbf{x}_i| p(\mathbf{x}_n, \mathbf{x}_i)$$

El indicador ρ_i^- representa el resultado de la merma de la capacidad retentiva de la memoria con respecto al estado emocional almacenado con índice i. Su valor es proporcional al indicador ρ_{i-1} de presencia del estado anterior y siempre menor que éste.

$$\rho_i^- \propto \rho_{i-1}$$
$$\rho_i^- < \rho_{i-1}$$

La variación descrita de los indicadores de presencia concuerda con la hipótesis de la curva del olvido [296] que dice que la capacidad retentiva de la memoria decae exponencialmente con respecto al tiempo y que esta disminución es menos acusada cuanto más reforzados estén los recuerdos. Este refuerzo es mayor a mayor intensidad emocional y mayor cantidad de repeticiones del hecho recordado.

Con todo ello la expresión del indicador ρ_i' se completa con las constantes α_1 y α_2 de proporcionalidad de la evocación y del olvido respectivamente.

$$\rho_i' = \alpha_1 \left|\mathbf{x}_i\right| p\left(\mathbf{x}_n, \mathbf{x}_i\right) + \alpha_2 \rho_{i-1}$$

Según la semántica del indicador ρ_i^- los valores posibles de la constante α_2 están acotados cumpliendo la siguiente condición:

$$0 < \alpha_2 < 1$$

Para garantizar la estabilidad del indicador ρ_i' evitando que éste crezca indefinidamente se debe cumplir que la suma de las constantes α_1 y α_2 no supere la unidad.

$$\alpha_1 + \alpha_2 \leq 1$$

De acuerdo con este criterio los valores posibles de la constante α_2 también están acotados cumpliendo la siguiente condición:

$$0 < \alpha_1 \leq 1 - \alpha_2$$

Gracias a ello todo indicador de presencia no supera en ningún caso el valor de intensidad máxima de los n estados emocionales almacenados en la memoria.

$$\forall \rho_i \ \left(\rho_i \leq \max \left\{ \left| \mathbf{x}_j \right| \ : \ 1 \leq j \leq n \right\} \right)$$

A su vez cada indicador de presencia ρ_i no supera en ningún caso su correspondiente valor $\rho_i^{\max}$.

$$\forall \rho_i \ \left(\rho_i \leq \rho_i^{\max} \ : \ \rho_i^{\max} = \alpha_1 \left| \mathbf{x}_i \right| + \alpha_2 \rho_{i-1}^{\max} \right)$$

Este valor permite expresar un indicador de precencia acotado de la siguiente manera:

$$0 \leq \frac{\rho_i}{\rho_i^{\max}} \leq 1$$

La función χ proporciona la estimación del estado emocional siguiente $\hat{\mathbf{x}}$ en base al estado de la memoria $\mathbf{b}$.

$$\begin{aligned} \chi : M^n &\rightarrow M^n \times E^\eta \\ \mathbf{b} &\rightarrow (\mathbf{b}, \hat{\mathbf{x}}) = \chi(\mathbf{b}) \end{aligned}$$

Esta función selecciona y devuelve el estado emocional almacenado $\mathbf{x}_i$ en la memoria $\mathbf{b}$ cumpliendo la siguiente expresión:

$$\chi(\mathbf{b}) = (\mathbf{b}, \mathbf{x}_i) \ \Big| \ \exists (\mathbf{x}_i, \rho_i) \in \mathbf{b} \ : \ \forall j \neq i \ , \ 1 \leq j \leq n \ , \ \frac{\rho_{i-1}}{\rho_{i-1}^{\max}} \left| \mathbf{x}_i \right| \geq \frac{\rho_{j-1}}{\rho_{j-1}^{\max}} \left| \mathbf{x}_j \right|$$

Otra utilidad del indicador de presencia asociado a cada estado emocional almacenado es la de marcar el orden de búsqueda yendo de mayor a menor, es decir, comenzando por los estados más presentes. Con ello es posible realizar búsquedas parciales, no completas, ahorrando recursos sin disminuir ostensiblemente la eficacia. Esto es debido a que por norma general es más probable encontrar un estado emocional entre aquellos con más presencia en un momento dado.

Para construir un estado emocional $\mathbf{x}$ a partir de un estado físico $\mathbf{s}$ definimos la función ϕ. Un estado físico es una tupla de λ números reales en la que cada uno se corresponde con el valor de una magnitud física.

$$\begin{aligned} \phi : \mathbb{R}^{\lambda} &\rightarrow E^{\eta} \\ \mathbf{s} &\rightarrow \mathbf{x} = \phi(\mathbf{s}) \end{aligned}$$

Esta función representa la percepción o generación de impulsos nerviosos a partir de los estímulos recibidos por el cuerpo del individuo. Es diferente en cada uno marcando ésta su naturaleza y en gran medida su conducta desde las tendencias innatas hasta el comportamiento más elaborado. La percepción, y por tanto esta función, es eminentemente estática, modificándose sólo por alteraciones bioquímicas, y su ajuste es el resultado de millones de años de evolución [297].

En psicofísica se estudia la relación entre la magnitud objetiva de un estímulo físico y la magnitud subjetiva de la sensación que produce. Normalmente esta relación no es lineal [298] debido a que la razón entre estímulo y sensación no se mantiene constante.

A partir del estudio de la sensibilidad la ley de Weber-Fechner [299] dice que la mínima variación apreciable de un estímulo es proporcional a su magnitud y que

por ello la relación estímulo-sensación es logarítmica. Por otro lado a partir del estudio directo de las estimaciones subjetivas de la magnitud de diferentes estímulos la ley de Stevens [300] dice que la relación estímulo-sensación es potencial con forma dependiente del exponente: lineal si es igual a 1, convexa si es mayor que 1 y cóncava si es menor que 1. Este exponente es diferente para cada tipo de estímulo.

La ley de Stevens es más general y verosímil matemáticamente [301] que la de Weber-Fechner pero requiere que el estímulo en cuestión sea simple y bien delimitado. En caso contrario muchas veces es factible seguir empleando la ley de Weber-Fechner.

Suponiendo que la magnitud de una sensación es proporcional a la cantidad de impulsos nerviosos que tiene asociada adoptamos la ley de Stevens para establecer la función ϕ en aquellas activaciones x_i originadas directamente por estímulos s_j.

$$\mathbf{x} = \phi(\mathbf{s})$$
$$x_i \propto s_j^{a_j} \ , \ a_j \in \mathbb{R}^+ \ , \ 1 \le i \le \eta \ , \ 1 \le j \le \lambda$$

En la Fig. 37 se muestran los tres elementos estructurales fundamentales de la arquitectura del proceso emocional-cognitivo desarrollado en este trabajo. Éstos son la *percepción*, la *memoria* y el *marco presente*. El metrónomo simboliza la naturaleza iterativa de este proceso el cual se ejecuta a una frecuencia determinada. Los cuadrados simbolizan variables, los rectángulos redondeados funciones y los círculos pesos.

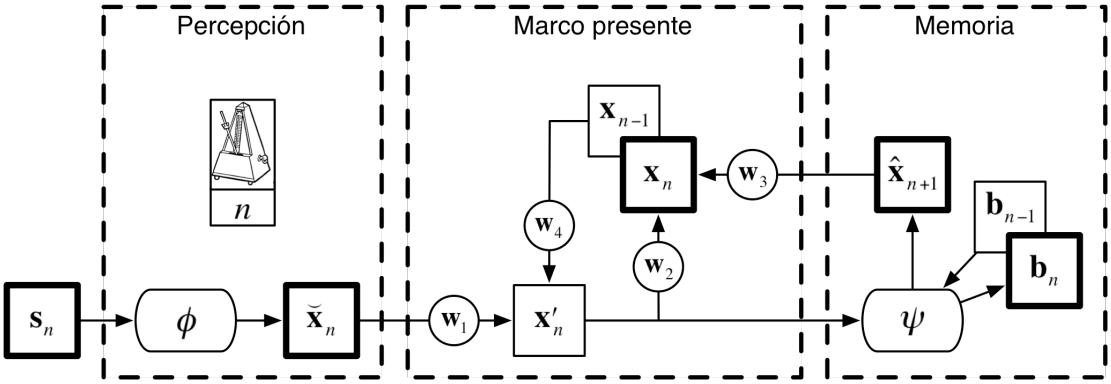

Fig. 37. Percepción, marco presente y memoria

El marco presente combina iterativamente las aportaciones de la percepción (parte no-cognitiva) $\breve{\mathbf{x}}_n$ y de la memoria (parte cognitiva) $\hat{\mathbf{x}}_{n+1}$ originando el estado emocional actual $\mathbf{x}_n$. Ejecutándo la función ϕ en el elemento percepción y la función ψ en el elemento memoria y siendo $\mathbf{w}_1$, $\mathbf{w}_2$, $\mathbf{w}_3$ y $\mathbf{w}_4$ diferentes pesos esta combinación queda descrita por las siguientes ecuaciones:

$$\breve{\mathbf{x}}_n = \phi(\mathbf{s}_n)$$
$$\mathbf{x}'_n = \mathbf{w}_1 * \breve{\mathbf{x}}_n + \mathbf{w}_4 * \mathbf{x}_{n-1}$$
$$(\hat{\mathbf{x}}_{n+1},\ \mathbf{b}_n) = \psi(\mathbf{x}'_n,\ \mathbf{b}_{n-1})$$
$$\mathbf{x}_n = \mathbf{w}_2 * \mathbf{x}'_n + \mathbf{w}_3 * \hat{\mathbf{x}}_{n+1}$$

El inicio de cada iteración viene marcado por un nuevo valor n. En primer lugar son combinados el estado emocional aportado por la percepción $\breve{\mathbf{x}}_n$ con el estado emocional anterior $\mathbf{x}_{n-1}$ dando lugar al estado emocional provisional $\mathbf{x}'_n$. A partir de éste la memoria aporta la estimación del estado emocional siguiente $\hat{\mathbf{x}}_{n+1}$. Finalmente la combinación del estado emocional provisional $\mathbf{x}'_n$ y de la estimación del estado emocional siguiente $\hat{\mathbf{x}}_{n+1}$ da lugar al estado emocional actual $\mathbf{x}_n$.

El estado emocional actual es esencialmente una predicción del futuro inmediato y su error e_n asociado se define como la distancia entre éste y el estado emocional siguiente proporcionado por la percepción.

$$e_n = d\left(\mathbf{x}_n, \breve{\mathbf{x}}_{n+1}\right)$$

3.6.3 Discusión

En 3.5.4 discutimos sobre todo la eficacia de los métodos estudiados (redes neuronales artificiales, algoritmos genéticos y sistemas de lógica difusa) identificando sus carencias y propusimos una serie de características para solventarlas consiguiendo un método cuasi-óptimo. Justificamos la no viabilidad de incorporar las nuevas características a los métodos estudiados y por ello definimos uno nuevo, en consonancia con los anteriores, que sí las cumple: el sistema emocional. Este sistema es un tipo de sistema cognitivo al ser un "sistema adaptativo complejo con capacidad de supervivencia que continuamente se encuentra en activo explorando y reaccionando autónomamente con el entorno" [302]. En este caso la capacidad de supervivencia se fundamenta en las emociones y la predicción del futuro inmediato.

En esta discusión nos centramos más en la eficiencia de los métodos estudiados comparados con nuestro sistema. Para ello es necesario extender los primeros para que acepten estados caracterizados por más de una variable. Al seguir un análisis multivariable los estados $\mathbf{s}_n$ se definen como vectores de λ variables dependientes entre sí.

$$\mathbf{s}_n = \left(s_{n1}, s_{n2}, \ldots, s_{n\lambda}\right)$$

Hay que puntualizar que las versiones de los métodos estudiados se encuentran entre las más simples que han sido útiles en problemas de predicción de sistemas caóticos deterministas. Éstas son el perceptrón multinivel con un solo nivel oculto como red neuronal artificial, el modelo AR ajustado por un algoritmo genético y el modelo Sugeno de orden cero como sistema de lógica difusa. Por ello presuponemos que en su categoría se encuentran también entre las más eficientes. Para evaluar la complejidad de cada método analizamos, en relación a λ y a m, el número de sinapsis de la red neuronal artificial, el número de parámetros a ajustar por el algoritmo genético y el número de coeficientes de los polinomios de las reglas del sistema de lógica difusa.

Las ecuaciones que definen al perceptrón multinivel con un nivel oculto y estados vectoriales son las siguientes:

$$c_k^n = \begin{cases} \tau_1\left(s_{n-m+k,1}\right) & , \; si \;\; 1 \le k \le m \\ \tau_2\left(s_{n-2m+k,2}\right) & , \; si \;\; m < k \le 2m \\ \ldots & \\ \tau_\lambda\left(s_{n-\lambda m+k,\lambda}\right) & , \; si \;\; (\lambda-1)m < k \le \lambda m \\ 0 & , \; si \;\; \lambda m < k \le q \end{cases}$$

$$a_k^n = \varphi_k\left(\sum_{j=1}^{q} w_{kj} a_j^{n-1} + b_k + c_k^n\right)$$

$$\hat{s}_{n+1,i} = \tau_i^{-1}\left(a_{q-\lambda+i}^n\right) \;,\; 1 \le i \le \lambda$$

El número q de neuronas y por tanto el ω de sinapsis aumenta. Siendo $q_E = \lambda m$ y $q_S = \lambda$ y suponiendo que $q_O \propto q_E$ se deduce que el nuevo número ω de sinapsis cumple la siguiente expresión:

$$\omega \propto (\lambda m)^2 + (\lambda m)(\lambda) = \lambda^2\left(m^2 + m\right)$$

Por tanto el número de sinapsis del perceptrón multinivel con un nivel oculto y estados vectoriales crece cuadráticamente con respecto a $\boldsymbol{\lambda}$ y a m. Esto hace que la complejidad de la red en la práctica sea muy elevada si los valores de $\boldsymbol{\lambda}$ y m no son pequeños. En la Fig. 38 se muestra un gráfico representativo del perceptrón multinivel descrito en el que se observa el aumento del número de neuronas $\boldsymbol{\lambda}$ veces.

Con estas condiciones la matriz $\mathbf{W}$ de este perceptrón multinivel con un solo nivel oculto es la siguiente:

$$\mathbf{W} = \begin{bmatrix}
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
w_{\lambda m+1,1} & w_{\lambda m+1,2} & \cdots & w_{\lambda m+1,\lambda m} & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
w_{\lambda m+2,1} & w_{\lambda m+2,2} & \cdots & w_{\lambda m+2,\lambda m} & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
w_{q-\lambda,1} & w_{q-\lambda,2} & \cdots & w_{q-\lambda,\lambda m} & 0 & 0 & \cdots & 0 & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & w_{q-\lambda+1,\lambda m+1} & w_{q-\lambda+1,\lambda m+2} & \cdots & w_{q-\lambda+1,q-\lambda} & 0 & 0 & \cdots & 0 \\
0 & 0 & \cdots & 0 & w_{q-\lambda+2,\lambda m+1} & w_{q-\lambda+2,\lambda m+2} & \cdots & w_{q-\lambda+2,q-\lambda} & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 0 & w_{q,\lambda m+1} & w_{q,\lambda m+2} & \cdots & w_{q,q-\lambda} & 0 & 0 & \cdots & 0
\end{bmatrix}$$

Las ecuaciones del modelo AR empleado con algoritmos genéticos con estados vectoriales, siendo $\mathbf{c}$ un vector de $\boldsymbol{\lambda}$ constantes y $\boldsymbol{\theta}_i$ cada matriz de parámetros, son las siguientes:

$$\hat{\mathbf{s}}_{n+1} = \mathbf{c} + \sum_{i=1}^{m} \boldsymbol{\theta}_i \mathbf{s}_{n+1-i}$$
$$\boldsymbol{\theta}_i \in \mathcal{M}_{\lambda \times \lambda}(\mathbb{R})$$

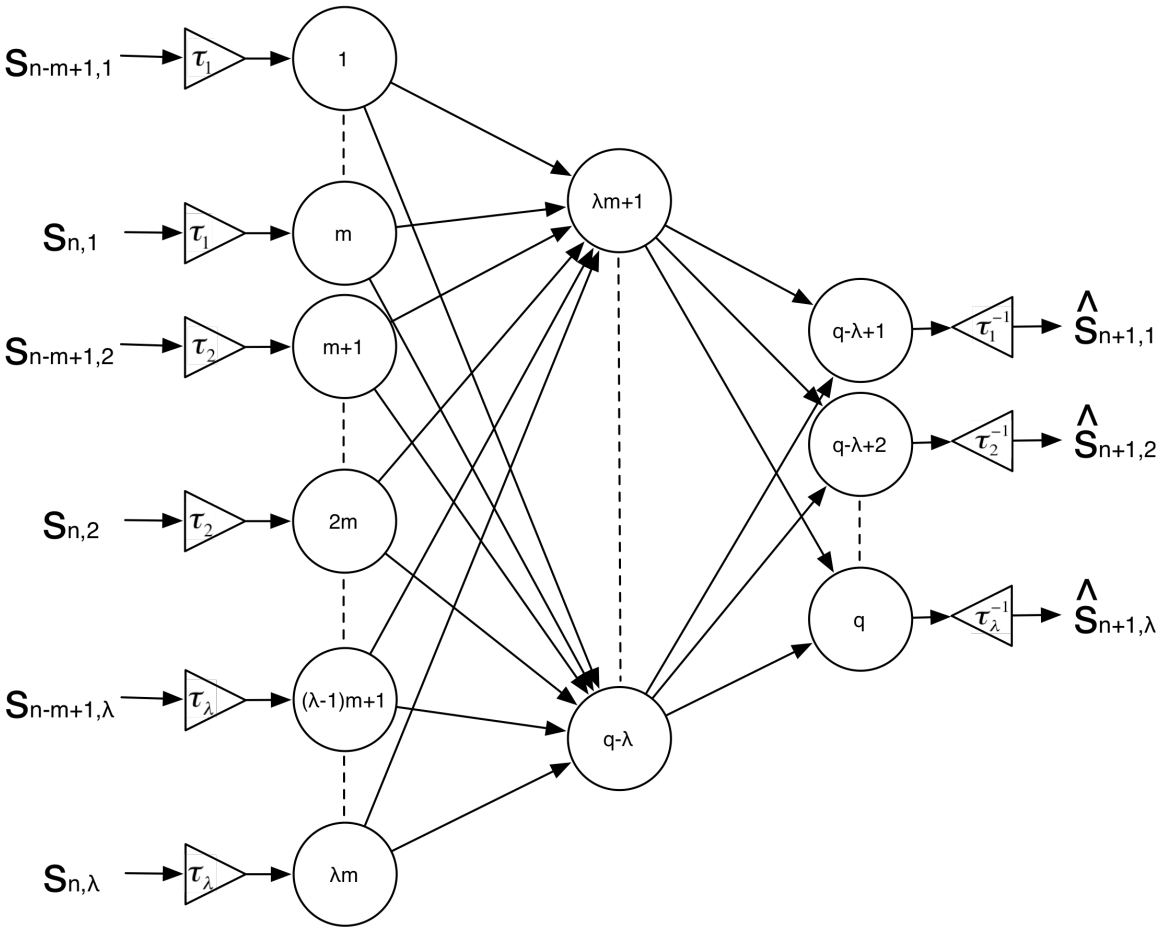

Fig. 38. Red neuronal artificial extendida

En este caso las matrices de parámetros $\boldsymbol{\theta}_i$ del modelo forman la matriz de matrices de parámetros $\boldsymbol{\Theta} \in \mathcal{M}_{1\times m}\left(\mathcal{M}_{\lambda\times\lambda}(\mathbb{R})\right)$. Su expresión es la siguiente:

$$\boldsymbol{\Theta} = \begin{bmatrix} \boldsymbol{\theta}_1 & \boldsymbol{\theta}_2 & \cdots & \boldsymbol{\theta}_m \end{bmatrix}$$

La misma expresión desarrollada tiene la forma:

$$\boldsymbol{\Theta} = \left[\begin{bmatrix} \theta_{1,1,1} & \theta_{1,1,2} & \cdots & \theta_{1,1,\lambda} \\ \theta_{1,2,1} & \theta_{1,2,2} & \cdots & \theta_{1,2,\lambda} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1,\lambda,1} & \theta_{1,\lambda,2} & \cdots & \theta_{1,\lambda,\lambda} \end{bmatrix} \begin{bmatrix} \theta_{2,1,1} & \theta_{2,1,2} & \cdots & \theta_{2,1,\lambda} \\ \theta_{2,2,1} & \theta_{2,2,2} & \cdots & \theta_{2,2,\lambda} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{2,\lambda,1} & \theta_{2,\lambda,2} & \cdots & \theta_{2,\lambda,\lambda} \end{bmatrix} \cdots \begin{bmatrix} \theta_{m,1,1} & \theta_{m,1,2} & \cdots & \theta_{m,1,\lambda} \\ \theta_{m,2,1} & \theta_{m,2,2} & \cdots & \theta_{m,2,\lambda} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{m,\lambda,1} & \theta_{m,\lambda,2} & \cdots & \theta_{m,\lambda,\lambda} \end{bmatrix} \right]$$

El número κ de parámetros crece linealmente con respecto a m y cuadráticamente con respecto a λ. Éste es igual a:

$$\kappa = \lambda^2 m$$

Un crecimiento tan rápido del número de parámetros a ajustar por un algoritmo genético es un problema debido a la ampliación exagerada del espacio de búsqueda que supone. Aunque este crecimiento es menor que el de sinapsis de la red neuronal para mismos valores de m y λ sigue siendo excesivo e incluso existe la posibilidad de que el algoritmo genético lo acuse mucho más que la red neuronal. Esto se debe a que el algoritmo de aprendizaje de la red neuronal artificial tiene un comportamiento determinista y acota su espacio de búsqueda de los pesos sinápticos óptimos mientras que un algoritmo genético tiene un comportamiento con cierto grado de aletoriedad y no acota tanto el espacio de búsqueda de los parámetros óptimos.

Las ecuaciones del modelo Sugeno de sistema de lógica difusa con estados vectoriales son las siguientes:

$$w_k = \min\left(\mu_{A_1^k}\left(s_{n-m+1,1}\right),\ \mu_{A_2^k}\left(s_{n-m+2,1}\right),\ \ldots,\ \mu_{A_{m+1}^k}\left(s_{n-m+1,2}\right),\ \ldots,\ \mu_{A_{\lambda m}^k}\left(s_{n,\lambda}\right)\right)$$

$$\mathbf{y}'_k = \boldsymbol{\gamma}^k$$

$$\hat{\mathbf{s}}_{n+1} = \frac{\sum_{i=1}^{q} w_i \mathbf{y}'_i}{\sum_{i=1}^{q} w_i}$$

Estructuralmente las modificaciones que sufre el sistema de lógica difusa al incorporar estados vectoriales son mínimas. El mayor cambio se centra en el contenido de la base de conocimiento. Aumenta el número de variables numéricas

de entrada de las reglas pasando a ser λm lo que hace que el número de reglas necesarias para contemplar todos los casos posibles también aumente.

El número υ de reglas difusas crece exponencialmente con respeto a λ y a m. Siendo l_j el número de términos lingüísticos asociados a la variable con índice j el número de reglas difusas es:

$$\upsilon = \prod_{j=1}^{\lambda} l_j{}^m$$

Este crecimiento tan elevado supone un coste considerable en la ejecución del sistema de lógica difusa. En cada iteración tienen que ser verificadas todas las reglas. También el algoritmo de aprendizaje de reglas difusas se ve afectado debido más al aumento del número de reglas que al cambio de forma vectorial. En estas condiciones la menor capacidad de generalización del sistema de lógica difusa hace que generalmente en comparación la red neuronal artificial se encuentre en posición de ventaja.

En este caso los vectores de coeficientes $\boldsymbol{\gamma}^k$ de los polinomios de los consecuentes de las reglas difusas forman la matriz de vectores de coeficientes $\boldsymbol{\Gamma} \in \mathcal{M}_{1\times\upsilon}\left(\mathcal{M}_{1\times\lambda}(\mathbb{R})\right)$. Su expresión es la siguiente:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\gamma}^1 & \boldsymbol{\gamma}^2 & \cdots & \boldsymbol{\gamma}^{\upsilon} \end{bmatrix}$$

La misma expresión desarrollada tiene la forma:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \begin{bmatrix} \gamma_1^1 & \gamma_2^1 & \cdots & \gamma_\lambda^1 \end{bmatrix} & \begin{bmatrix} \gamma_1^2 & \gamma_2^2 & \cdots & \gamma_\lambda^2 \end{bmatrix} & \cdots & \begin{bmatrix} \gamma_1^{\upsilon} & \gamma_2^{\upsilon} & \cdots & \gamma_\lambda^{\upsilon} \end{bmatrix} \end{bmatrix}$$

El número ι de coeficientes, al igual que el número υ de reglas difusas, crece exponencialmente con respeto a λ y a m. Éste es igual a:

$$\iota = \lambda\upsilon = \lambda\prod_{j=1}^{\lambda} l_j^{\,m}$$

Resumiendo, con los mismos valores de λ y m se cumple que:

1. $\kappa < \omega << \iota$, siendo κ la cantidad de incognitas del algoritmo genético, ω la cantidad de incognitas de la red neuronal artificial y ι la cantidad de incognitas del sistema de lógica difusa.

2. $\pi_{\mathbf{W}}(\omega) < \pi_{\mathbf{\Theta}}(\kappa)$ y $\pi_{\mathbf{W}}(\omega) < \pi_{\mathbf{\Gamma}}(\iota)$, siendo $\pi_{\mathbf{W}}(\omega)$ la complejidad asociada a la resolución de las ω incognitas de la red neuronal artificial, $\pi_{\mathbf{\Theta}}(\kappa)$ la complejidad asociada a la resolución de las κ incognitas del algoritmo genético y $\pi_{\mathbf{\Gamma}}(\iota)$ la complejidad asociada a la resolución de las ι incognitas del sistema de lógica difusa.

3. Siendo $\pi_{\mathbf{W}}(\omega)$ estrictamente creciente con ω y elevada para valores pequeños de ω, llegando a ser impracticable con facilidad, deducimos que ninguno de estos métodos es adecuado.

Nuestro sistema ha sido diseñado para no usar ninguna matriz de incognitas como la $\mathbf{W}$ de la red neuronal artificial, la $\mathbf{\Theta}$ del algoritmo genético o la $\mathbf{\Gamma}$ del sistema de lógica difusa, las cuales especifican la estructura del método dependiendo fuertemente de λ y m. Las incognitas de estas matrices marcan el comportamiento del algoritmo empleado y son resueltas utilizando la secuencia de estados. Nuestro sistema opera directamente con la secuencia de estados prescindiendo de este tipo de matrices.

La variación de λ no afecta estructuralmente a nuestro sistema al operar directamente con vectores y no sólo con escalares como ocurre con los métodos

anteriores (redes neuronales artificiales, algoritmos genéticos y sistemas de lógica difusa).

El valor m no se contempla en nuestro método. En los otros métodos para predecir el estado siguiente usamos los últimos m estados. En cambio con el nuestro usamos sólo el último estado y una memoria interna (de tamaño n y por tanto reducido) que almacena la secuencia de estados completa. En los otros métodos utilizar sólo el último estado implica un $m = 1$ (valor mínimo posible) y utilizar la secuencia de estados completa un $m = n$ (valor máximo posible). Como afirmamos en 3.5.4, una disminución de m suele mejorar la eficiencia ($min(t) \wedge min(C)$) a costa de la eficacia ($min(e)$), y viceversa, siempre y cuando no se llegue a valores extremos los cuales hacen impracticable el método en cuestión. Nuestro sistema carece de este problema y no sufre este enfrentamiento entre eficiencia y eficacia.

En nuestro sistema cada estado físico es almacenado en la memoria como un estado emocional con su correspondiente indicador de presencia. Los indicadores de presencia sirven tanto para ponderar estados emocionales como para recoger información contextual. Éstos varían según el contexto provocando que un estado emocional más presente tenga más probabilidades de ser evocado.

En contraposición con los métodos estudiados, nuestro sistema integra la predicción y el aprendizaje en un solo algoritmo. En cada iteración la memoria como base de conocimiento se reconfigura aumentando indefinidamente y actualizando los indicadores de presencia de los estados emocionales almacenados. Esto no afecta a la complejidad del algoritmo al estar ordenada la memoria y no precisar que sea exhaustivo para dar resultados aceptables.

Finalmente podemos concluir que:

1. $\varepsilon << \kappa$ y $\varepsilon = 0$, siendo ε la cantidad de incognitas de nuestro método. Esto quiere decir que éste no tiene incognitas que resolver. Además también es independiente de λ y m lo que lo hace plenamente escalable.

2. $\pi_{\Psi}(\varepsilon) < \pi_{\mathrm{W}}(\omega)$, siendo $\pi_{\Psi}(\varepsilon)$ la complejidad asociada al proceso de aprendizaje de nuestro método y teniendo en cuenta que la resolución de incognitas de los métodos estudiados es a su vez la materialización de sus respectivos procesos de aprendizaje.

3. Siendo $\pi_{\Psi}(\varepsilon)$ baja y estando unificado en nuestro método el proceso de predicción-aprendizaje destaca su eficiencia frente a la de los métodos estudiados. Este proceso también es el más sencillo de implementar y su mayor parte se reduce básicamente a la actualización del indicador de presencia (operación aritmética simple) de cada uno de como máximo los n estados emocionales almacenados en la memoria.

Capítulo 4

Resultados experimentales

Resumen: este capítulo incluye los resultados de las pruebas practicadas para demostrar la eficacia del nuevo método desarrollado: el sistema emocional. Estos resultados se analizan a nivel funcional y de rendimiento para evaluar también la eficiencia del nuevo método en el entorno real.

4.1 Escenarios

La comprobación de un algoritmo de predicción suele realizarse comparando una muestra de datos recogida previamente con los resultados obtenidos a partir de ella. Este procedimiento presenta algunos problemas en general y sobre todo cuando se opera en tiempo real. Éstos son:

1. Evaluación exclusiva de la eficacia. Un algoritmo de predicción puede ser más o menos eficaz pero una eficiencia mínima es siempre un requisito obligado. Además en muchos casos debe tenerse en cuenta la interrelación existente en la práctica entre eficacia y eficiencia.

2. Carácter excesivamente sintético. Al encontrarse el algoritmo de predicción totalmente aislado no se ve afectado por sucesos accidentales típicos de una ejecución real como son los fallos en la recepción de datos o la carencia momentánea de recursos computacionales.

3. Ausencia de retroalimentación. Una muestra de datos recogida antes de la ejecución del algoritmo de predicción sólo es válida si las variables de entrada de éste son independientes en todo momento de los resultados obtenidos.

Estos inconvenientes hacen que estas pruebas tengan una reproducibilidad limitada afectando al nivel de confianza de las conclusiones obtenidas a partir de ellas. Por ello es más fructífero realizarlas in situ mediante ejecuciones reales en tiempo real.

En nuestro caso particular probar in situ el sistema emocional integrado en los agentes software plantea algunas dificultades. Básicamente éstas son:

1. Una vez predicha una interrupción grave y suspendido el *video-streaming* no es posible determinar con qué antelación se ha actuado. Esto se debe a que al suspender el *video-streaming* las variables observables se reducen al RSSI con lo que la predicción no es confirmable posteriormente.

2. No es posible determinar la demora en la reanudación del *video-streaming* tras la mejora del estado del canal inalámbrico. El motivo es que no es confirmable el estado del *video-streaming* estando éste suspendido.

Estos contratiempos los solventamos aprovechando el hecho de que si el punto de acceso se mantiene estático existe un espacio físico alrededor de él en el que es posible ejecutar el *video-streaming* sin interrupciones graves y que su forma aunque cambiante es prácticamente estacionaria. En el espacio físico exterior o complementario a éste no es posible ejecutar el *video-streaming*. El momento en que se cruza la frontera entre estos espacios es el óptimo para suspender o reanudar el *video-streaming* según el caso.

Una vez localizada la frontera repetimos varios patrones de movimiento cada uno con una trayectoria y velocidad propia. En la Fig. 39 se muestran esquemáticamente los seis patrones utilizados. La circunferencia simboliza la citada frontera y los cuadrados y círculos los puntos de inicio y fin de cada movimiento respectivamente.

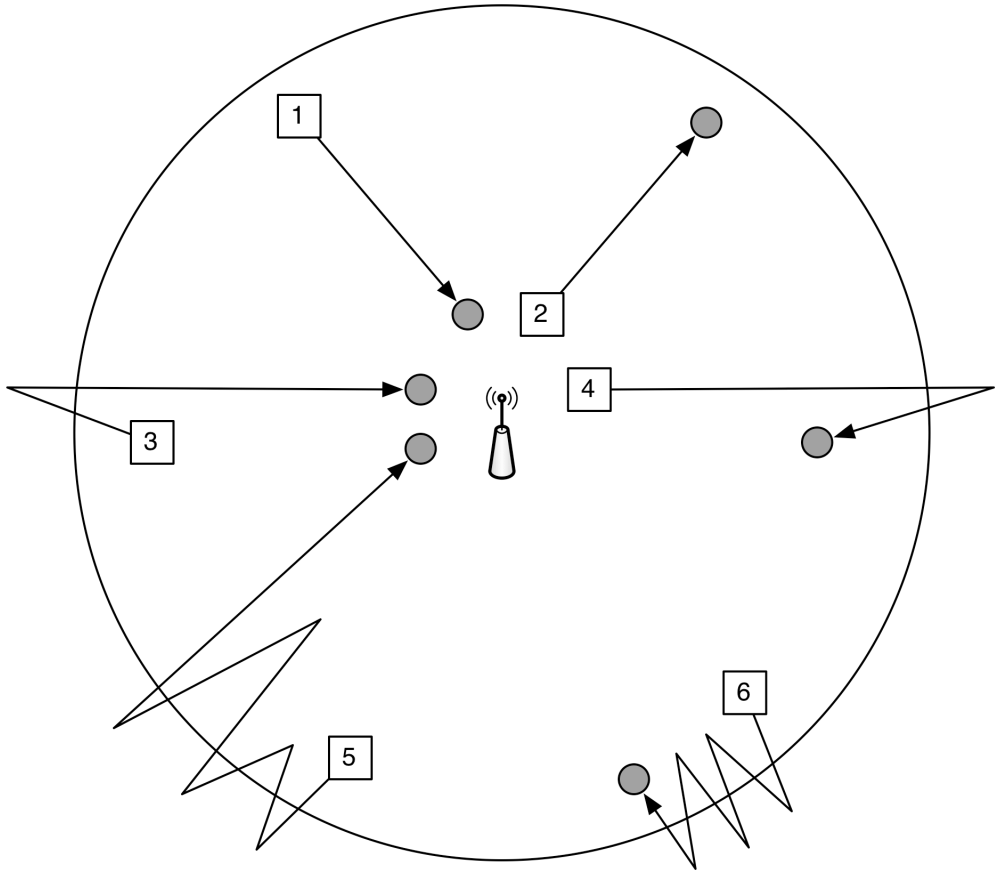

Fig. 39. Patrones de pruebas

4.2 Herramientas

El hardware empleado para realizar las pruebas es un computador portátil como servidor de *video-streaming* y proxy-servidor, un teléfono móvil inteligente como cliente de *video-streaming* y proxy-cliente, y un punto de acceso WiFi. En detalle éstos son:

- Computador portátil: MacBook Air 13'' Early 2014 [303] (Intel Core i5-4260U [304] de doble núcleo a 1,4 GHz y Turbo Boost de hasta 2,7 GHz, 8 GB RAM, 256 GB Flash) con OS X Yosemite v10.10.3 [305] y Oracle Java SE 8u45 [306].

- Teléfono móvil inteligente: Samsung Galaxy Ace 4 SM-G357FZ [65] (Quad-Core 1,2 GHz, 1GB RAM) con Android 4.4.4 KitKat [64] (modelo para desarrollo), Motorola Moto G (3rd Gen.) [67] (Quad-Core 1,4 GHz, 1GB RAM) con Android 5.1.1 Lollipop [66] (modelo para desarrollo) y Samsung Galaxy Ace 2 GT-I8160P [63] (Dual-Core 800 MHz, 768 MB RAM) con Android 4.1.2 Jelly Bean [62] (modelo para pruebas finales).

- Punto de acceso WiFi: Cisco WAP200 Wireless-G Access Point [307].

Cambiando en las pruebas finales el teléfono móvil inteligente por uno de menos prestaciones con una versión de Android más exigente en recursos computacionales perseguimos recrear un escenario más adverso y con mayor necesidad de eficiencia por parte de la solución desarrollada.

Los servidores de *streaming* empleados en el desarrollo de la solución son el Darwin Streaming Server 6.0.3 [308] (software libre) y el QuickTime Broadcaster 1.5.3 [309] (software gratis) pero para las pruebas finales se han sustituido por el

Wowza Streaming Engine 4.1.0 [120] (software propietario) y el Telestream Wirecast 5.0.4 [310] (software propietario) respectivamente, con licencias ofrecidas por el fabricante para su evaluación temporal. Estos últimos se encuentran entre las alternativas comerciales más potentes y difundidas del momento en su categoría.

La plataforma de agentes software sobre la que se sustenta la solución desarrollada en este trabajo es la JADE [311]. Ésta es un proyecto de software libre fundado por *Telecom Italia* y está completamente escrita en Java. Sus agentes cumplen con la especificación *Foundation for Intelligent Physical Agents* (*FIPA*) [312] y pueden ser distribuidos en diferentes máquinas con distintos sistemas operativos. La variante de JADE compatible con dispositivos móviles es la denominada *Light Extensible Agent Platform* (*LEAP*) [313] la cual mantiene la misma *Application Programming Interface* (*API*) que su original. La versión concreta utilizada de JADE y JADE-LEAP es la 4.3.3 [314].

Existen otras plataformas de agentes software, similares a JADE, escritas también en Java, libres y que cumplen el estándar FIPA. Algunas de ellas son Zeus [315], Aglets [316] y *Secure Mobile Agents* (*SeMoA*) [317]. Zeus fue desarrollada inicialmente por *British Telecom* y está especialmente orientada al desarrollo de la capacidad de razonamiento basado en reglas de los agentes. Aglets en sus comienzos fue un producto comercial de *IBM* y su principal virtud es la capacidad de movilidad fuerte de sus agentes. SeMoA fue desarrollada por la *Fraunhofer-Gesellschaft* y está enfocada sobre todo a mejorar la seguridad de los agentes.

La elección de JADE-LEAP frente a otras plataformas se justifica por su compatibilidad con los dispositivos móviles capaces de ejecutar Java o .NET [318] y las facilidades que presenta incorporando mecanismos orientados a mejorar las

comunicaciones entre agentes en redes inalámbricas. También es interesante su metodología de desarrollo propia [319] integrable con herramientas como el *Unified Modeling Language* (*UML*) [320] o el *Agent Unified Modeling Language* (*AUML*) [321] entre otras.

JADE-LEAP tiene dos modos de ejecución: *Stand-alone* y *Split*. En el modo *Stand-alone* un contenedor completo se ejecuta en el dispositivo móvil. En el modo *Split* el contenedor se divide en un *FrontEnd*, que se ejecuta en el dispositivo móvil, y un *BackEnd*, que se ejecuta en un servidor remoto, comunicados permanentemente tal como se muestra en la Fig. 40.

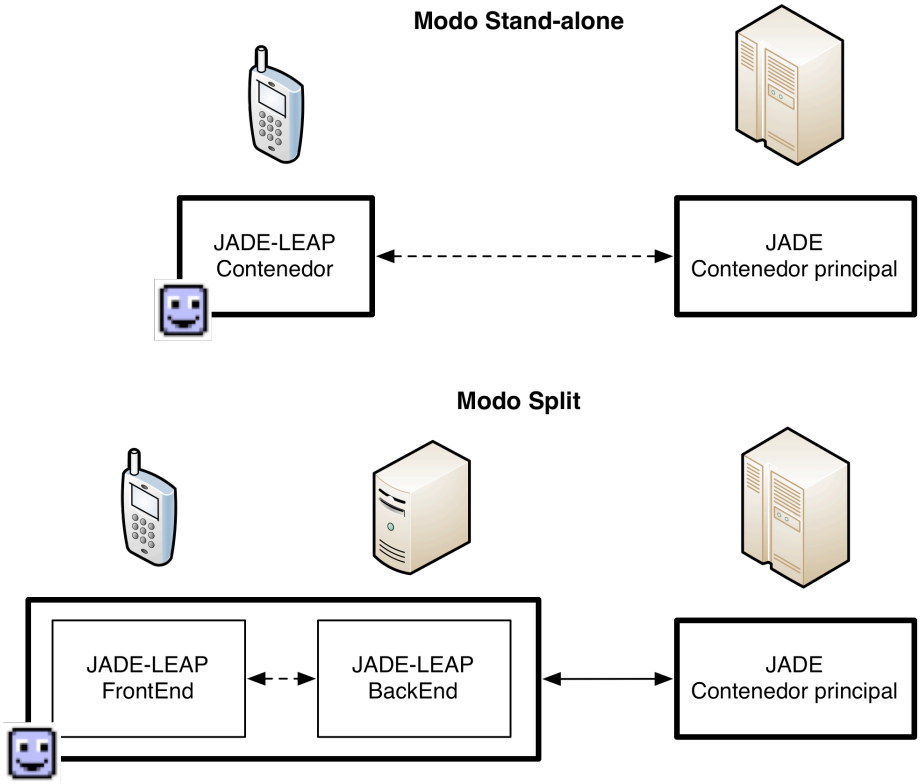

Fig. 40. Modos de ejecución de JADE-LEAP

El modo de ejecución *Split* está especialmente diseñado para aprovechar mejor los limitados recursos de los dispositivos móviles y para gestionar las desconexiones de la red inalámbrica entre el *FrontEnd* y el *BackEnd*. Cuando se produce una desconexión no se pierden mensajes. En este caso éstos son recibidos por el *BackEnd* el cual los comunica posteriormente al *FrontEnd* una vez reconectado automáticamente.

Otras bibliotecas usadas además de JADE y JADE-LEAP son una revisión de la implementación de RTP, *JLibRTP 0.2.2* [322], una revisión de la implementación de SDP, *jSDP 1.1* [323] y la biblioteca de generación de gráficos complejos *JFreeChart 1.0.19* [324].

Las herramientas de desarrollo empleadas son el analizador de tráfico *Wireshark 1.12.4* [325], el *Entorno de Desarrollo Integrado* (*IDE*) *Eclipse Luna SR2 (4.4.2)* [326] y el editor de ontologías *Protégé 3.3.1* [327] con el plugin *OntologyBeanGenerator 3.1* [328] para la generación de ontologías tipo JADE.

El análisis de los datos originados en las pruebas se ha apoyado en el lenguaje de programación y entorno de computación estadística *R 3.2.1* [329] acompañado del IDE *RStudio Desktop 0.99* [330].

4.3 Implementación

La implementación de la solución desarrollada completamente en Java sigue el esquema mostrado en 2.7.2 e ilustrado por la Fig. 19. En este caso el lenguaje de comunicación de los agentes es el *Agent Communication Language* (*ACL*) [331] de FIPA, lenguaje heredero del *Knowledge Query Manipulation Language* (KQML) [332].

Para reducir la carga computacional asociada al agente JADE-LEAP del proxy-cliente optamos por limitar sus responsabilidades llegando a comportarse éste como un agente reactivo simple. Además, para su ejecución empleamos el modo *Split*. El agente JADE del proxy-servidor aguanta la mayor carga computacional y se comporta como una combinación de agente reactivo simple y agente emocional.

Para mitigar los problemas de encaminamiento de paquetes UDP que se dan en la comunicación interna de datos del proxy establecemos que sea siempre el proxy-cliente quien la inicie. Esto requiere que sea conocida la dirección IP y el puerto UDP de datos del proxy-server. Con el primer paquete el proxy-cliente se identifica y el proxy-server asocia al proxy-cliente la dirección y puerto de origen de ese paquete. A partir de ese momento el proxy-server envía los datos destinados al proxy-cliente a esa dirección y puerto. Ésta en algunos casos es la original del proxy-cliente y en otros la de un encaminador. Este procedimiento evita que sea necesario que el proxy-cliente y el proxy-server sean accesibles directamente o que el proxy-cliente disponga de una dirección pública.

El proxy tiene tres modos de ejecución según el tipo de contenido y las preferencias del usuario. En cada uno el comportamiento al suspender el *video-streaming* ante una interrupción es diferente. Estos modos son:

1. *Video-streaming bajo demanda*. El proxy-cliente y el proxy-servidor detienen el *video-streaming* por separado mediante el método RTSP PAUSE.

2. *Video-streaming en vivo*. El proxy-servidor siempre almacena el *streaming* en su buffer pero éste sólo es dispuesto si no existe interrupción. El retraso producto de cada interrupción se acumula en la reproducción.

3. *Video-streaming en vivo en tiempo real*. En esta modalidad hay dos reproductores. Cuando no hay interrupción uno de ellos recibe el *video-streaming* directamente mientras que el otro recibe el *video-streaming* almacenado en el buffer tras el transcurso de la última interrupción.

La configuración del proxy se realiza mediante un archivo de constantes en texto plano en el que se especifican sus parámetros. Éstos están agrupados en parámetros de monitorización, de sensores, de receptores, del marco presente y de la memoria. En la Fig. 41 se muestra un ejemplo ilustrativo de este archivo, el cual es el utilizado en las pruebas.

En la monitorización del *video-treaming* el agente proxy-cliente proporciona periódicamente al agente proxy-servidor la información recopilada por los sensores establecidos. El agente proxy-servidor analiza emocionalmente esta información y decide en cada momento si suspender temporalmente el *streaming* ante una interrupción presumiblemente incipiente o reanudarlo si se estima posible.

```
# Monitor
monitor.period= 1000
monitor.timeout= 2000
monitor.verbose= false

# Sensors
sensor.id=                  RSSI,           DELAY,          JITTER,         PACKET_LOSS
sensor.qualityThreshold=    ,               400,            150,            0.05

# Receptors
receptor.id=            RSSI,           DELAY,          JITTER,         PACKET_LOSS
receptor.lowerLimit=    -96,            0,              0,              0.00
receptor.upperLimit=    -32,            400,            150,            0.05
receptor.zeroPoint=     -44,            50,             10,             0.00
receptor.kNeg=          -40000,         100,            50,             100
receptor.aNeg=          1.5,            1.0,            1.0,            1.0
receptor.kPos=          100,            -10000,         -5000,          -20000
receptor.aPos=          1.0,            1.0,            1.0,            0.5

# Present Frame
presentFrame.perceptionWeight= 1.0

# Memory
memory.evocationConstant= 0.33
```

Fig. 41. Archivo de configuración del proxy

En el archivo de configuración el período de muestreo del agente proxy-cliente expresado en ms se corresponde con el parámetro `monitor.period`. El parámetro `monitor.timeout` indica en ms el tiempo de espera máximo del agente proxy-servidor por los informes de monitorización del agente proxy-cliente. Una vez transcurrido este tiempo se considera inoperativo el agente proxy-cliente por desconexión de su *FrontEnd*. El parámetro `monitor.verbose` señala si los informes de monitorización son exhaustivos. En este caso además de la información proporcionada por los sensores se incorpora una lista de metadatos de cada paquete del *video-streaming* recibido por el proxy-cliente. Esta opción es ineficiente y está orientada sólo a labores de depuración.

En el proxy se establece un `monitor.period` de 1000 ms. Éste se encuentra en un punto de equilibrio entre eficacia (mejor con períodos menores) y eficiencia (mejor con períodos mayores). El valor de `monitor.timeout` es el doble del de `monitor.period`. Evidentemente `monitor.verbose` queda desactivado.

Los umbrales de QoD indican los límites aceptables de retraso, jitter y pérdida de paquetes. Si éstos son sobrepasados hay peligro de pérdida ostensible e inadmisible de QoE lo que obliga a suspender el *video-streaming* previamente. En el archivo de configuración estos umbrales se establecen por medio del parámetro `sensor.qualityThreshold`.

Los valores de los umbrales de QoD establecidos para el proxy son los discutidos en 2.6. Éstos son 400 ms para el retraso, 150 ms para el jitter y 5 % para la pérdida de paquetes. Para el RSSI no se indica ningún umbral porque en la práctica no tiene una relación definida y estable con la QoD.

Los parámetros de los receptores del archivo de configuración del proxy determinan la función ϕ del modelo emocional. Cada receptor i origina una

activación x_i a partir del estímulo s_i. Este estímulo es perceptible en el rango de medida $\left[l_i^{\inf}, l_i^{\sup}\right]$ cuyos extremos se corresponden con los parámetros `receptor.lowerLimit` y `receptor.upperLimit` respectivamente.

En el proxy el rango de medida para el RSSI se encuentra entre -96 dBm y -32 dBm, y el del resto de receptores entre 0 y su correspondiente umbral de QoD.

Para simplificar la definición de los receptores admitimos que cada uno origine activaciones de distinto signo. El valor del estímulo a partir del cual la activación cambia de signo lo llamamos *punto cero* o_i el cual se corresponde con el parámetro `receptor.zeroPoint`. Para este valor la activación es nula.

$$l_i^{\inf} \leq o_i \leq l_i^{\sup}$$

El punto cero marca la situación que no supone ni una amenaza ni una ventaja. Si éste no es acertado el agente emocional experimenta un predominio injustificado de emociones positivas o negativas que le hace tergiversar la realidad y en consecuencia tomar decisiones incorrectas. Por ello es necesario elegirlo con cuidado.

Para el proxy hemos fijado el `receptor.zeroPoint` en -44 dBm de RSSI, 100 ms de retraso, 10 ms de jitter y 0 % de pérdida de paquetes debido a la pésima tolerancia a la pérdida de paquetes del reproductor de vídeo nativo de Android en todas las versiones con las que hemos trabajado. En este caso, con un punto cero tan extremo, se da una preponderancia de emociones negativas lo cual da un carácter conservador.

Correspondiéndose k_i^- con el parámetro `receptor.kNeg`, k_i^+ con el parámetro `receptor.kPos`, a_i^- con el parámetro `receptor.aNeg` y a_i^+ con el parámetro `receptor.aPos`, en este caso la función ϕ se define de la siguiente manera:

$$\mathbf{x} = \phi(\mathbf{s})$$

$$x_i = \begin{cases} k_i^- \left(\dfrac{|s_i - o_i|}{|l_i^{\inf} - o_i|} \right)^{a_i^-} & , \; si \;\; s_i < o_i \\ 0 & , \; si \;\; s_i = o_i \\ k_i^+ \left(\dfrac{|s_i - o_i|}{|l_i^{\sup} - o_i|} \right)^{a_i^+} & , \; si \;\; s_i > o_i \end{cases} \quad , \quad k_i^- k_i^+ < 0 \;,\; a_i^- > 0 \;,\; a_i^+ > 0$$

La realimentación del modelo emocional es configurable modificando los pesos $\mathbf{w}_1$ y $\mathbf{w}_4$. El valor de cada elemento del peso $\mathbf{w}_1$ es igual al indicado por el parámetro `presentFrame.perceptionWeight`. El peso $\mathbf{w}_4$ cumple la siguiente expresión:

$$\mathbf{w}_4 = \mathbf{1} - \mathbf{w}_1$$

Las constantes α_1 y α_2 de proporcionalidad de la evocación y del olvido también son determinadas estáticamente. El valor de α_1 es igual al indicado por el parámetro `memory.evocationConstant`. La constante α_2 cumple la siguiente expresión:

$$\alpha_2 = 1 - \alpha_1$$

Para ajustar estos valores de los parámetros de los receptores, marco presente y memoria es necesario seguir un procedimiento evolutivo. Cada ajuste provoca una personalidad distinta en el agente emocional. La óptima varía en cada situación y en general se busca una equilibrada. Este procedimiento es posible acelerarlo teniendo en cuenta la influencia de los cambios de cada parámetro en el sistema emocional final.

Los coeficientes k_i indican el signo y la intensidad máxima de la activación producto del receptor. Activaciones mayores favorecen evocaciones más episódicas,

como ocurre en las mujeres comparadas con los hombres [333]. También las activaciones mayores tienen más influencia en el estado emocional. En un momento dado una activación muy intensa limita la capacidad de atención al hacer despreciables las demás. Este fenómeno ocurre en animales [334] como mecanismo de optimización de recursos. Si es injustificado puede llegar a ser una limitación y una amenaza en sí mismo. Un desequilibrio permanente puede llevar al agente emocional a padecer el equivalente a un trastorno del espectro autista.

Los exponentes a_i marcan la distribución de la sensibilidad de los receptores. A partir del punto cero si $a_i = 1$ la sensibilidad es constante, si $a_i < 1$ es elevada y decreciente y si $a_i > 1$ es reducida y creciente. De este modo se puede optar por ser más sensible a los estímulos que están más cercanos al punto cero o al límite del rango de medida.

Siendo constantes los parámetros de los receptores se evita que el agente emocional pueda padecer el equivalente a un trastorno bipolar o el efecto de drogas psicoactivas.

En el proxy las activaciones positivas producto de los receptores son exíguas y su sensibilidad en este caso constante. Las activaciones negativas tienen una mayor intensidad siendo la correspondiente al jitter la más pequeña, seguida del retraso, la pérdida de paquetes y el RSSI. Por este orden el valor de cada una es el doble de la anterior. La sensibilidad en este caso es constante para el retraso y el jitter, mayor para los estímulos cercanos al punto cero para la pérdida de paquetes y mayor para el límite del rango de medida para el RSSI.

Por simplicidad se ha optado por no usar realimentación en el proxy lo que se consigue dando al parámetro `presentFrame.perceptionWeight` un valor de 1. Esto hace que no se tengan en cuenta las predicciones pasadas para obtener las

siguientes, o lo que es lo mismo, que el agente emocional no prediga un futuro más lejano. El sistema emocional opera con un período igual al período mínimo de muestreo. Al ser este tiempo suficiente para reaccionar sólo es necesario predecir exactamente el estado siguiente y no más allá.

En animales la realimentación añade una complejidad adicional porque el cerebro procesa los objetos reales e imaginarios de la misma forma, estando la percepción y la memoria integradas en el mismo mecanismo cognitivo [335]. Esto, muchas veces provoca conflictos y, al manejar un modelo de la realidad y no la realidad en sí, son los objetos con una carga emocional mayor los que prevalecen. El mantener la coherencia de este modelo a toda costa se torna una necesidad más de supervivencia [336] que de conocer la realidad. Llevada al extremo, buscando sólo limitar el pensamiento y conseguir emociones positivas en el corto plazo, suele ser perniciosa manifestándose entre otras formas en: adicciones, prejuicios, fanatismos, convicciones, ideologías o religiones.

Las constantes de proporcionalidad de la evocación y del olvido marcan el comportamiento paradójico de la memoria: recordar causa olvido. En este caso el olvido tiene un carácter adaptativo para solucionar la competencia entre recuerdos [337] producto de la evocación. La evocación es la base de todos los procesos mentales que implican la memoria comenzando éstos inconscientemente, tanto los simples como los complejos [338]. Una mayor influencia de la evocación implica también un mayor olvido lo que obliga a equilibrarlos. Un desequilibrio de éstos puede llevar a un pensamiento desordenado que para el agente emocional supone padecer el equivalente a una esquizofrenia hebefrénica.

En el proxy el valor del parámetro `memory.evocationConstant` es de un tercio lo que proporciona un equilibrio adecuado entre evocación y olvido en la memoria.

Finalmente, para solventar el problema que supone la desaparición de determinados tipos de estímulo al suspenderse el *streaming*, los cuales no son captados por los receptores, definimos tres estados diferentes en los que puede encontrarse el sistema emocional:

- *Vigilia*: si todos los tipos de estímulo son captados por los receptores. El sistema emocional opera con normalidad para determinar cuando es necesario suspender el *streaming*.

- *Sueño*: si los receptores sólo captan el estímulo correspondiente al RSSI por encontrarse el *streaming* suspendido. En consonancia con el estado de sueño en animales el sistema emocional no genera nuevos recuerdos pero sí modifica los existentes [339] según la percepción parcial que se mantiene. La evocación de recuerdos se vuelve más volátil al comparar sólo parte de las activaciones y no todas ellas. En este caso las predicciones se tienen en cuenta para decidir cuando reanudar el *streaming*.

- *Congelación*: si ningún estímulo es captado por los receptores al encontrarse inoperativo el agente proxy-cliente por desconexión de su *FrontEnd*. El sistema emocional se detiene.

4.4 Pruebas experimentales

La validación del proxy desarrollado se realiza en sus tres modos de ejecución (bajo demanda, en vivo y en vivo en tiempo real) con las herramientas citadas en 4.2 y con recursos externos de prueba como [340], el cual es comunicado por un servidor de *streaming* Wowza desplegado en una instancia de Amazon Elastic Compute Cloud (Amazon EC2) [341].

Tras una validación positiva del proxy continuamos con las pruebas finales del sistema emocional. Para una mayor homogeneización de éstas se han realizado con vídeo bajo demanda. El vídeo empleado es el de prueba del servidor de streaming Wowza el cual tiene las siguientes características:

- *Resolución*: 424x240.
- *Formato vídeo*: H.264 – MPEG-4 AVC (part 10).
- *Tasa de fotogramas*: 24.
- *Formado audio*: MPEG AAC Audio (mp4a).
- *Tasa de muestreo*: 48000 Hz (estéreo).
- *Velocidad*: 520 kbps.

Las condiciones ambientales en las que se han hecho las pruebas en exterior y las características del movimiento del dispositivo móvil son las siguientes:

- *Temperatura*: 23º C.
- *Humedad:* 70%.

- *Velocidad*: 0,75 mps.
- *Forma*: línea recta.

Y para las mismas pruebas realizadas en interior:

- *Temperatura*: 24º C.
- *Humedad:* 65%.
- *Velocidad*: 0,75 mps.
- *Forma*: línea recta.

El entorno exterior es una calle recta de una zona residencial, sin obstáculos y con cinco redes WiFi domésticas operando simultáneamente (señal débil). El punto de acceso se sitúa en la puerta de una casa (27,920794º N, 15,434353º O).

El entorno interior se encuentra en la sede del Departamento de Ingeniería Telemática (DIT) de la ULPGC (28,071376º N, 15,453116º O). Es un pasillo de 60 m de largo por 2,5 m de ancho, paredes anchas de 60 cm, con cristaleras y escaleras en sus extremos, y despachos y laboratorios de telecomunicaciones en sus laterales. El punto de acceso se sitúa en la mitad del pasillo en uno de los despachos, de 4 m de largo por 2,5 m de ancho, con la puerta abierta y a 1 m de ésta. Hay dos puntos de acceso WiFi (señal fuerte) y multitud de usuarios y aparatos que provocan interferencias frecuentes y acusadas.

Para determinar el error cometido en cada predicción empleamos el error e_n asociado al estado emocional actual que definimos en 3.6.2. Éste es un error absoluto que al tener unidad de medida resulta poco práctico para realizar

comparaciones. Por ello definimos el error relativo e'_n, el cual es adimensional, de la siguiente manera:

$$e'_n = \frac{e_n}{\left\|\breve{\mathbf{x}}_{n+1}\right\|} = \frac{d\left(\mathbf{x}_n, \breve{\mathbf{x}}_{n+1}\right)}{\left\|\breve{\mathbf{x}}_{n+1}\right\|}$$

Para situar cada uno de los patrones ilustrados en la Fig. 39 marcamos una serie de puntos comunes en el espacio geográfico:

- Punto cercano al cercano al punto de acceso: P_{PA}.
- Punto medio entre el punto de acceso y la frontera: P_M.
- Punto cercano a la frontera: P_F.
- Punto en la frontera: P_X.
- Punto de desconexión física más allá de la frontera: P_D.

El punto P_X tanto en exterior como en interior se encuentra en un intervalo amplio; la frontera oscila en exterior 5 m y en interior 3 m con picos esporádicos cercanos al punto P_{PA}. Para reducir el número de estos picos ha sido necesario reiniciar el punto de acceso frecuentemente.

Cada uno de los patrones de movimiento citados se puede describir inequívocamente como una sucesión de los puntos marcados. Los resultados para cada patrón comenzando con la memoria vacía en cada intento se explican a continuación.

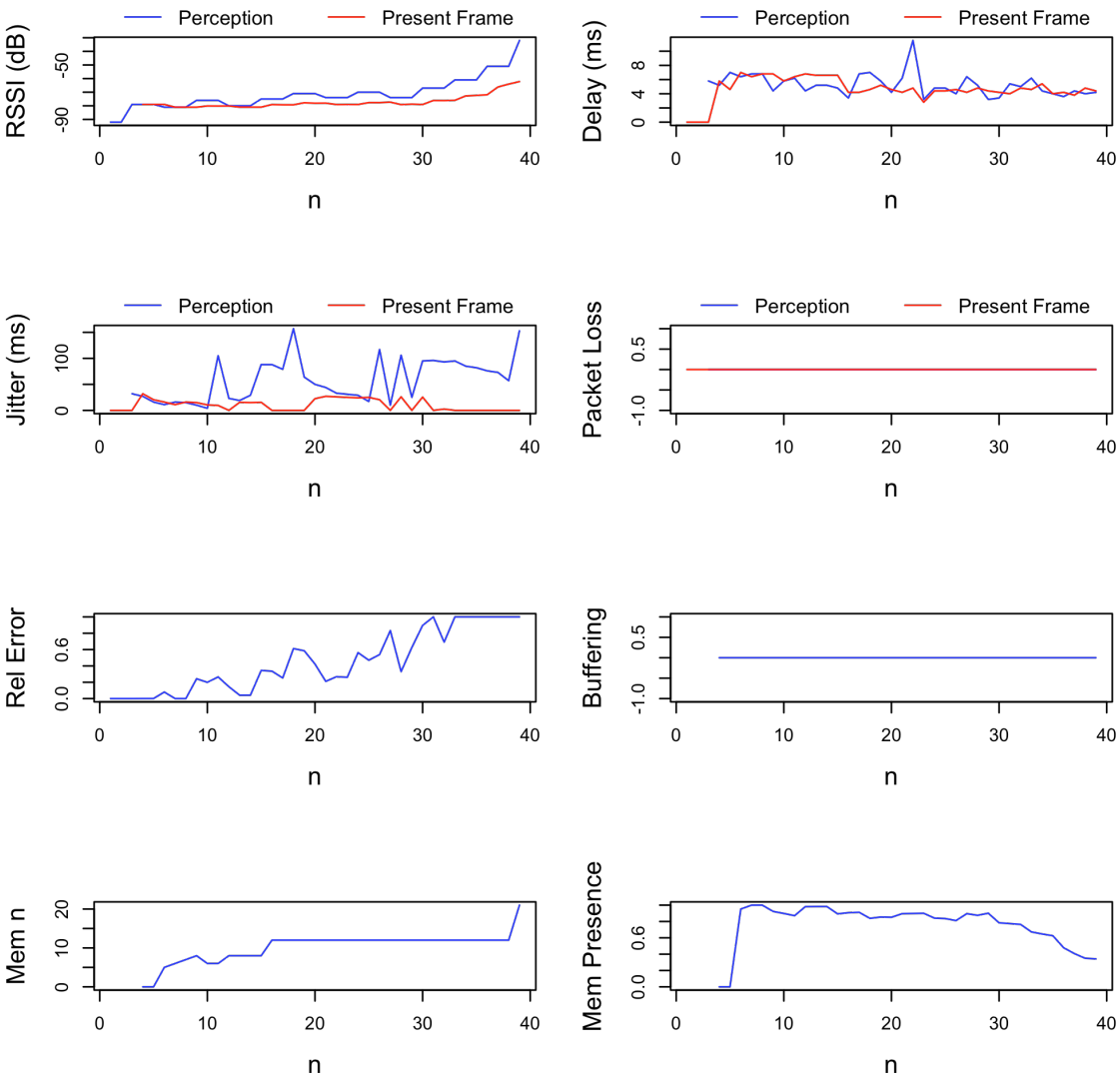

Fig. 42. Parámetros de rendimiento del patrón 1 (exterior)

Patrón 1: *teléfono móvil acercándose al punto de acceso* ($P_F \rightarrow P_{PA}$)

La Fig. 42 muestra los parámetros de rendimiento en valores físicos al seguir el patrón 1. En las gráficas con dos líneas se combina lo percibido (*Perception*) con la estimación del estado siguiente (*Present Frame*).

La gráfica de RSSI muestra el carácter conservador (sesgo emocional negativo pronunciado) que se le ha dado al método de predicción del proxy mediante el

ajuste de sus parámetros. Éste tarda tiempo en adquirir confianza al aumentar el RSSI y por eso cuando está cerca del punto de acceso comete más error (por inercia). Sin embargo, sigue siendo un error pequeño en valores físicos.

En cuanto al retraso (*Delay*), se observa que en las primeras iteraciones aparece y se estabiliza en un valor bajo. Después de $n = 20$ hay un ínfimo incremento, que no se explica por una bajada de RSSI, y sin embargo éste es recuperado volviendo a la media. Al ser este incremento esporádico y muy pequeño es despreciado por el método.

El jitter estimado muestra siempre un desfase de aproximadamente 2 o 3 iteraciones, y siempre captura la tendencia; pero en una escala de 0 a 10 en lugar de la escala 0 a 100 que tiene el jitter percibido. Las estimaciones del jitter son menos precisas por ser el estímulo con menos peso en el estado emocional.

No se produce pérdida de paquetes (*Packet Loss*).

Para el error relativo (*Rel Error*) se observa que éste es menor cuando las condiciones son más adversas con un RSSI bajo.

No se llega a hacer *buffering* porque no se produce ninguna desconexión.

En cuanto al estado evocado (*Mem n*) se observa que prácticamente es siempre el mismo entre las iteraciones $n = 15$ y $n = 35$ porque en éstas el estado emocional percibido prácticamente no varía.

En cuanto a la presencia del estado evocado (*Mem Presence*) no cambia entre $n = 7$ y $n = 30$ porque los recuerdos existentes son similares a lo que se percibe. Sin embargo, a partir de ese intervalo comienza a disminuir porque lo percibido es novedoso y diferente en comparación con esos recuerdos.

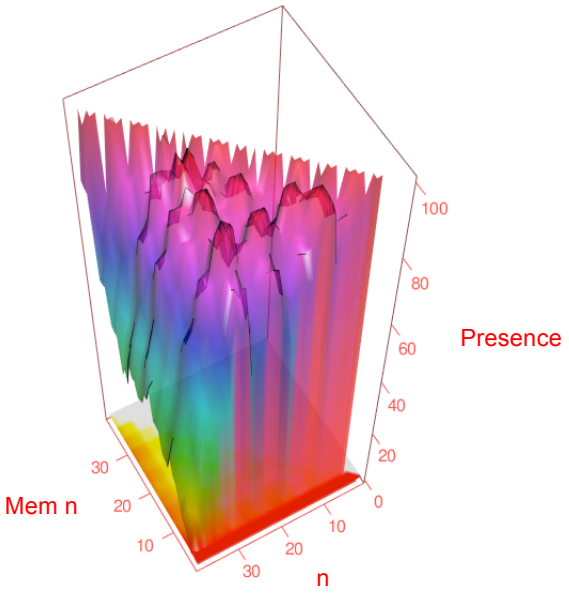

Fig. 43. Evolución de la memoria con el patrón 1 (exterior)

En la Fig. 43 se muestra la evolución de la memoria al seguir el patrón 1. Esta representación siempre tiene forma de prisma triangular cuya base es un triángulo isósceles rectángulo. El eje n indica la iteración del método, el eje *Mem n* la iteración en la que fue almacenado cada estado en la memoria y el eje *Presence* la presencia de cada estado en la memoria. La forma triangular se produce porque en cada iteración se añade un estado emocional a la memoria. Los colores de la gráfica van desde el rojo (menor presencia) al violeta (mayor presencia).

En este caso se observa la forma acusada de prisma plano hasta cerca de $n = 35$ porque hasta ese momento el estado emocional percibido no varía acusadamente. A partir de ese momento, al aparecer estados diferentes y de menor intensidad a cualquiera de los contenidos en la memoria, se produce una disminución generalizada de la presencia de cada estado. Esto se aprecia en la cara izquierda del prisma por la aparición de colores azules y verdes.

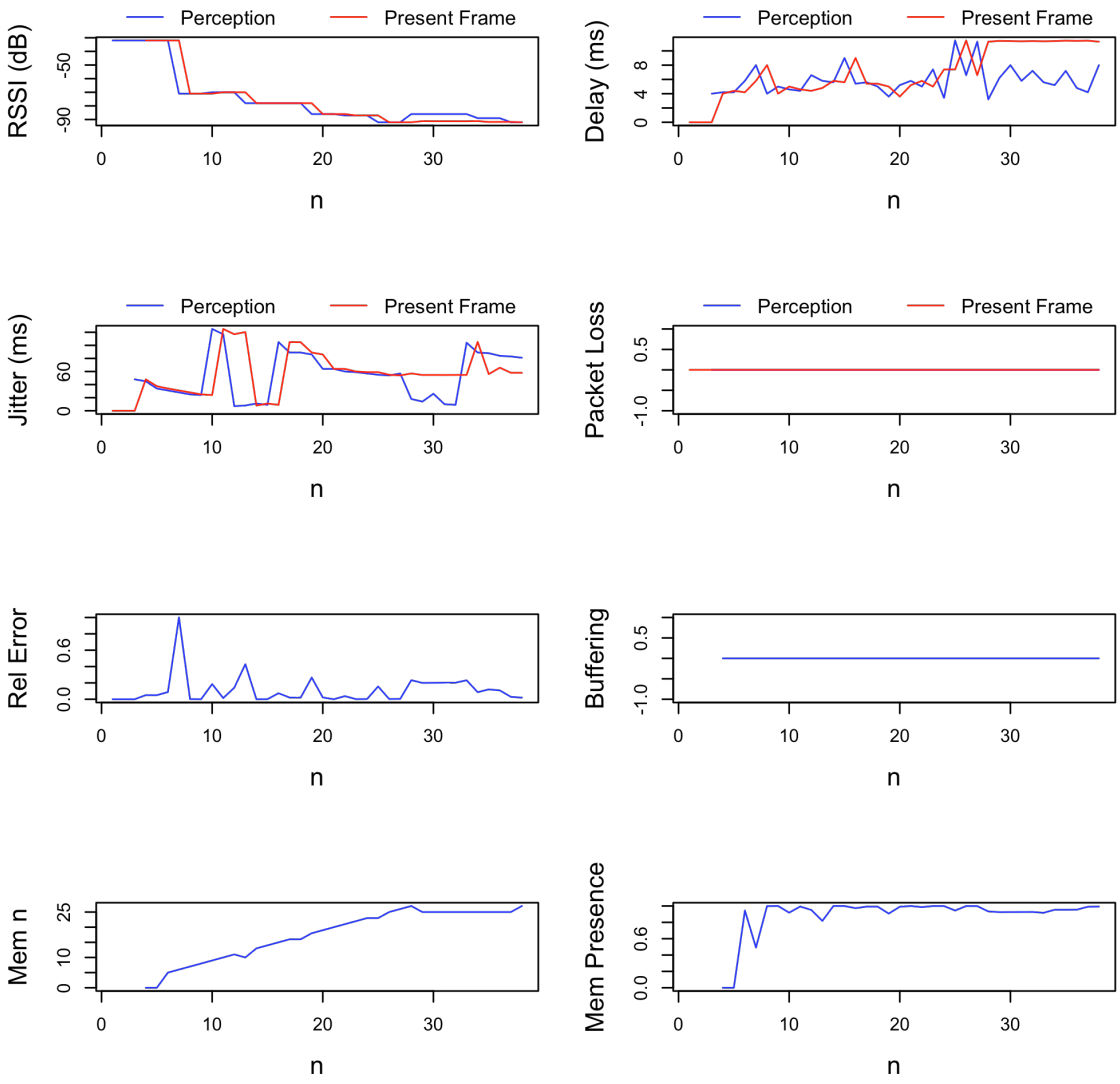

Fig. 44. Parámetros de rendimiento del patrón 2 (exterior)

Patrón 2: *teléfono móvil alejándose del punto de acceso* ($P_{PA} \rightarrow P_F$)

En la Fig. 44 la gráfica de RSSI muestra una estimación con muy poco error, lo que vuelve a mostrar el carácter conservador que se le ha dado al método, cometiendo menos error cuando el RSSI empeora que cuando mejora.

En cuanto al retraso, al igual que en el patrón anterior aparece en las primeras iteraciones y se estabiliza. A continuación se encuentran unas oscilaciones

cortas y una mayor y prolongada a partir de $n = 25$. Aunque el retraso vuelve a estabilizarse el método pierde la confianza y estima un valor mayor.

El jitter vuelve a tener un desfase de aproximadamente 2 o 3 iteraciones capturando la tendencia pero en este caso situándose en la misma escala. Al ir empeorando la situación la estimación es más acertada.

No se produce pérdida de paquetes.

Con el error relativo se observa claramente que éste es inferior cuando las condiciones son adversas por su disminución cuando éstas van empeorando.

No se llega a hacer *buffering*.

En cuanto al estado evocado se observa que hasta superar la iteración $n = 25$ es siempre reciente mánteniendose a continuación cuando el estado emocional percibido deja de variar.

La presencia del estado evocado desde que aumenta en las primeras iteraciones se mantiene elevada en toda la prueba. Ésta se ve favorecida por ser siempre los nuevos estados de igual o mayor intensidad que los anteriores.

En la Fig. 45 se muestra la evolución de la memoria al seguir el patrón 2. En este caso la forma de prisma es más redondeada predominando los estados más recientes sobre los más antiguos que van siendo olvidados hasta que el estado percibido deja de variar. Esto se aprecia en la cara derecha del prisma por la aparición de colores azules y verdes, además de por su forma.

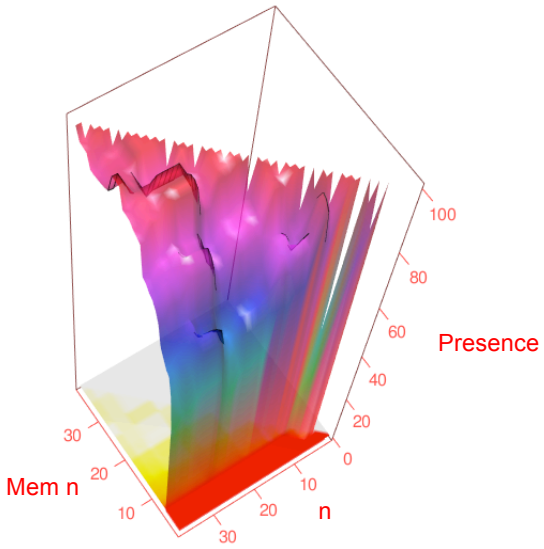

Fig. 45. Evolución de la memoria con el patrón 2 (exterior)

Patrón 3: *teléfono móvil desconectándose, reconectándose y acercándose al punto de acceso* ($P_F \rightarrow P_D \rightarrow P_{PA}$)

Este patrón se puede descomponer en: patrón 2 (empezando en P_F) + desconexión + patrón 1 (ampliado).

En la Fig. 46 se observa que el RSSI sigue el mismo comportamiento que en el patrón 2 hasta que aparece la desconexión. Una vez reconectado sigue el mismo comportándose que en el patrón 1, con poca confianza.

La estimación del retraso, el jitter y la pérdida de paquetes es muy ajustada pero a pesar de ello no se consigue predecir la desconexión a tiempo. Al reconectar se observa en el jitter y la pérdida de paquetes que se va recuperando la confianza.

El error relativo es bajo hasta que aumenta abruptamente tras la reconexión al perder la confianza.

En este caso sí se llega a hacer *buffering*.

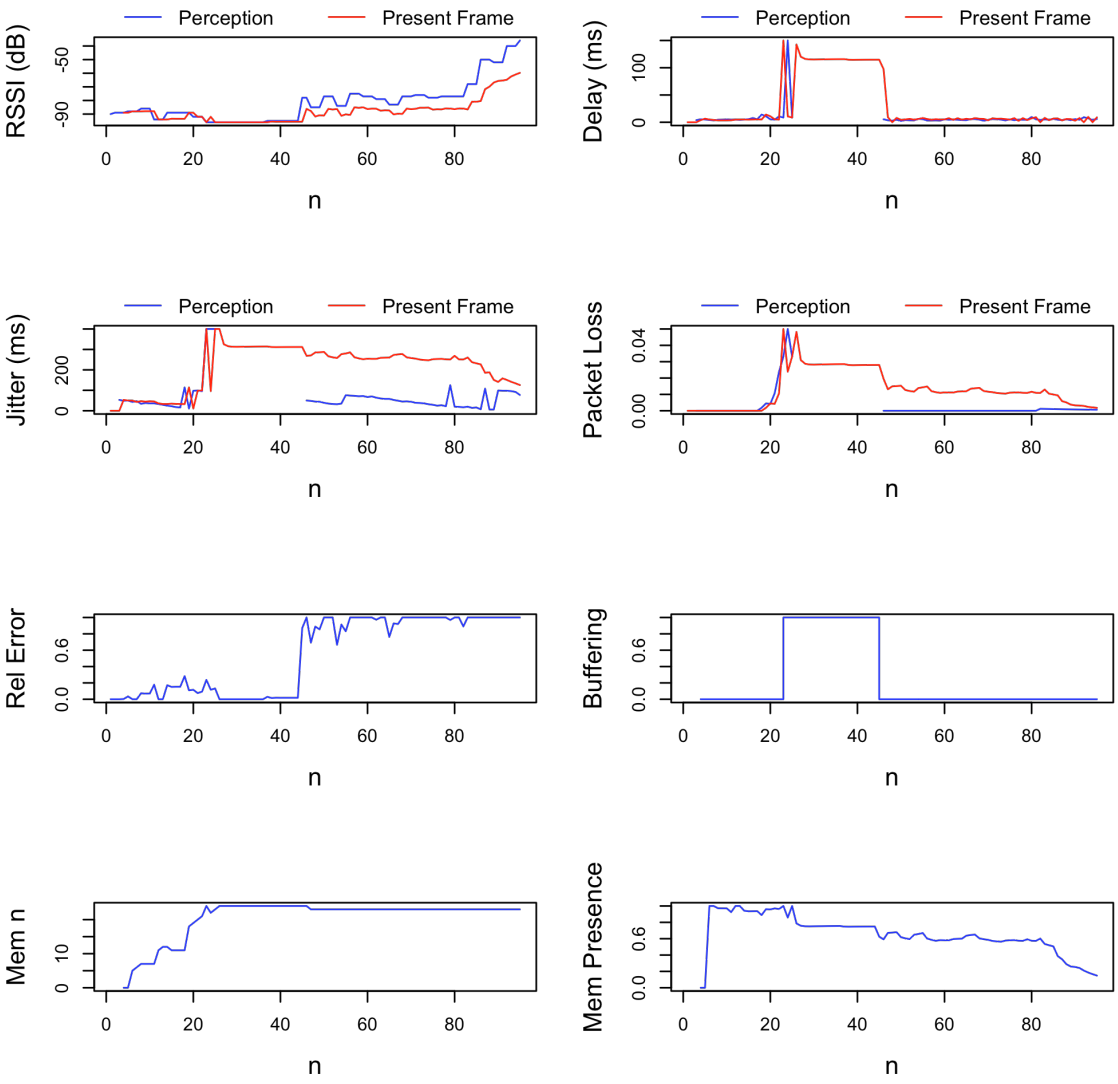

Fig. 46. Parámetros de rendimiento del patrón 3 (exterior)

En cuanto al estado evocado se observa que hasta la desconexión está entre los recientes. A partir de la reconexión mantiene uno cercano al momento de la desconexión como un trauma del que se va recuperando lentamente.

La presencia del estado evocado es elevada antes de la desconexión, como en el patrón 2, y va disminuyendo a partir de la reconexión, como en el patrón 1.

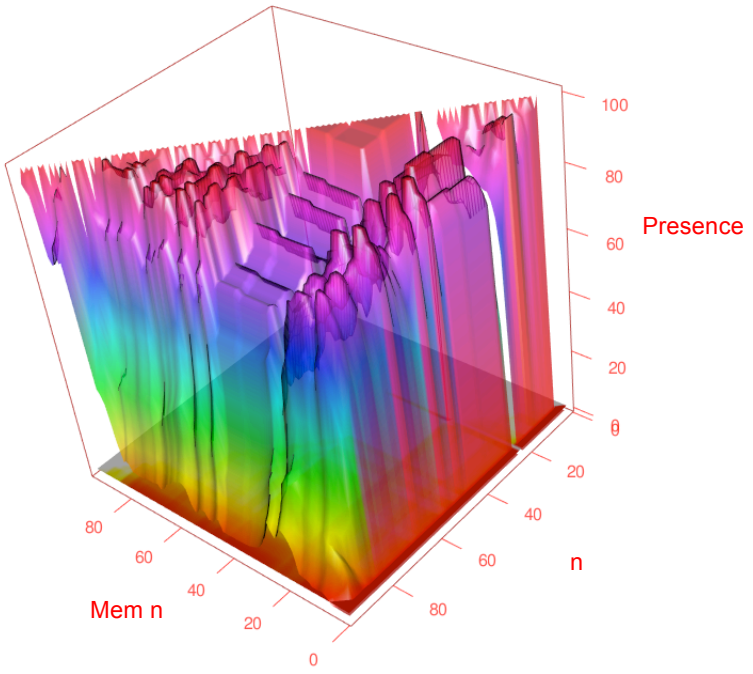

Fig. 47. Evolución de la memoria con el patrón 3 (exterior)

En la Fig. 47 llama la atención un espacio en blanco. Éste se corresponde con el momento de la desconexión. A partir de éste aparece una meseta plana originada en el estado de sueño. A continuación al reconectar la figura guarda gran similitud con la del patrón 1.

Patrón 4: *teléfono móvil alejándose del punto de acceso, desconectándose y reconectándose* ($P_{PA} \rightarrow P_D \rightarrow P_F$)

Este patrón es similar al anterior. Se puede descomponer en: patrón 2 (ampliado) + desconexión + patrón 1 (terminando en P_F). Su evolución es la misma salvo que la desconexión se produce más tarde.

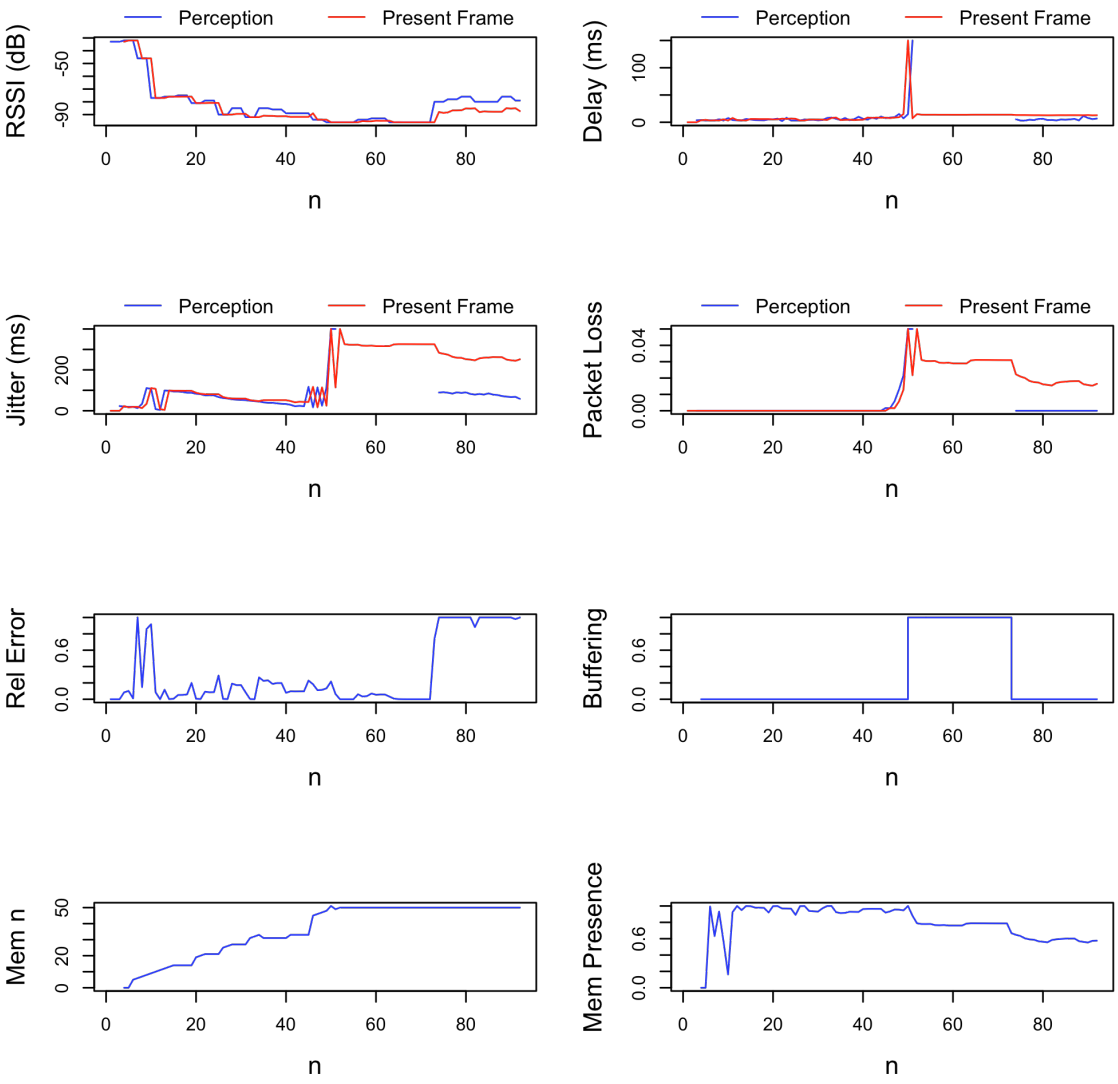

Fig. 48. Parámetros de rendimiento del patrón 4 (exterior)

La Fig. 48 es muy similar a la Fig. 46 salvo que está desplazada. El comportamiento de todas las variables es el mismo. Tampoco se consigue predecir la desconexión a tiempo.

Sólo es destacable el pico de error relativo y de presencia del estado evocado en torno a $n = 10$ por la abrupta caída del RSSI.

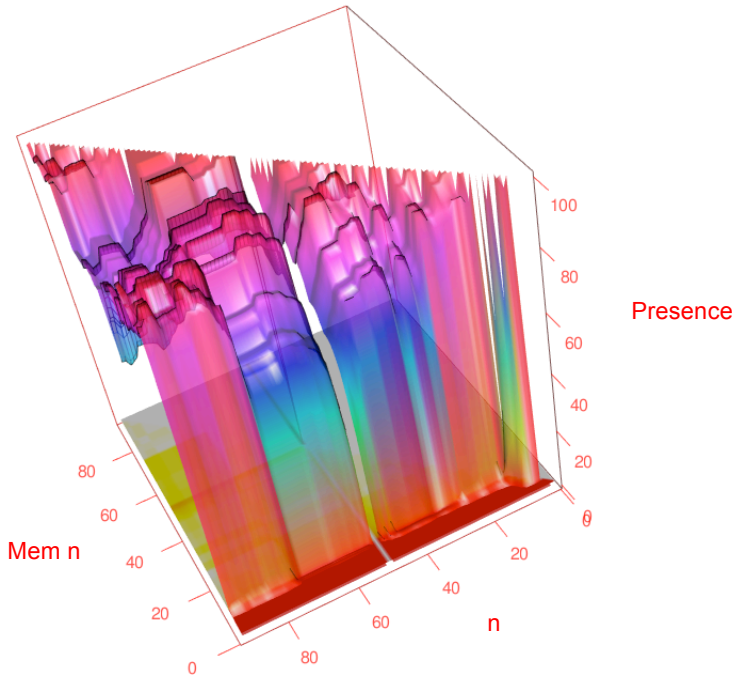

Fig. 49. Evolución de la memoria con el patrón 4 (exterior)

La Fig. 49 es parecida a la Fig. 47. Se aprecia con más claridad que la superficie antes de la desconexión es similar a la del patrón 2. Al reconectar ésta sigue teniendo una forma redondeada pero va volviéndose más plana hasta las últimas iteraciones.

Patrón 5: *tres desconexiones acercándose cada vez más al punto de acceso* ($P_F \rightarrow P_D \rightarrow P_F \rightarrow P_D \rightarrow P_M \rightarrow P_D \rightarrow P_{PA}$)

Este patrón es más complejo: se producen tres desconexiones. Se puede descomponer en: patrón 2 + desconexión + patrón 1 + patrón 2 + desconexión + patrón 1 + patrón 2 + desconexión + patrón 1.

En la Fig. 50 se aprecian los patrones anteriores y el mismo comportamiento conservador. En este caso la primera desconexión no se consigue predecir a tiempo pero las siguientes sí y en el intervalo en el que se sitúa el punto P_X . Se producen varios *buffering* porque las condiciones en la frontera son oscilantes como se deduce a partir del RSSI.

En la Fig. 51 también se aprecia los patrones anteriores, además de las zonas de desconexión y de sueño. Se ve claramente que acaba como el patrón 1.

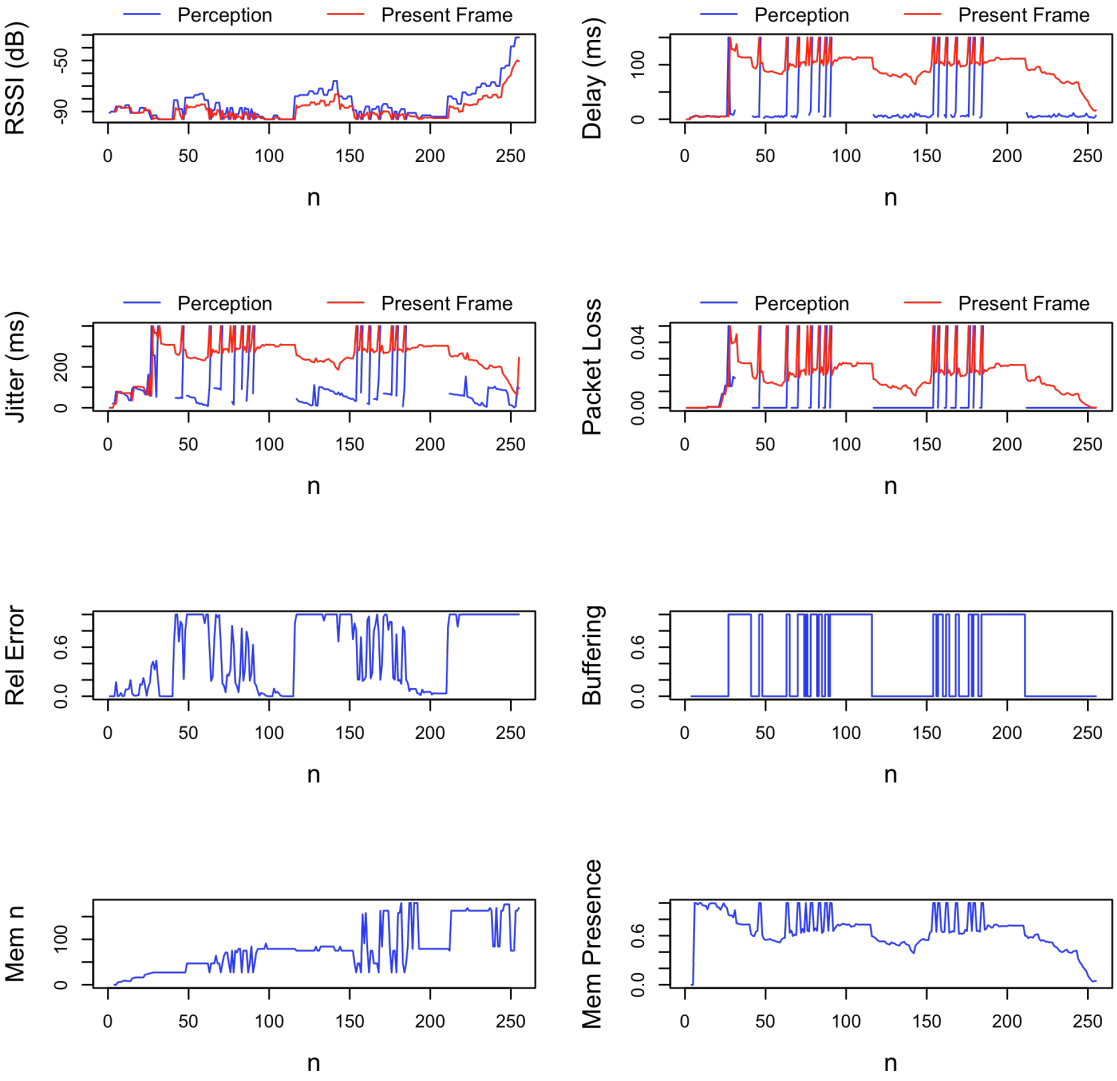

Fig. 50. Parámetros de rendimiento del patrón 5 (exterior)

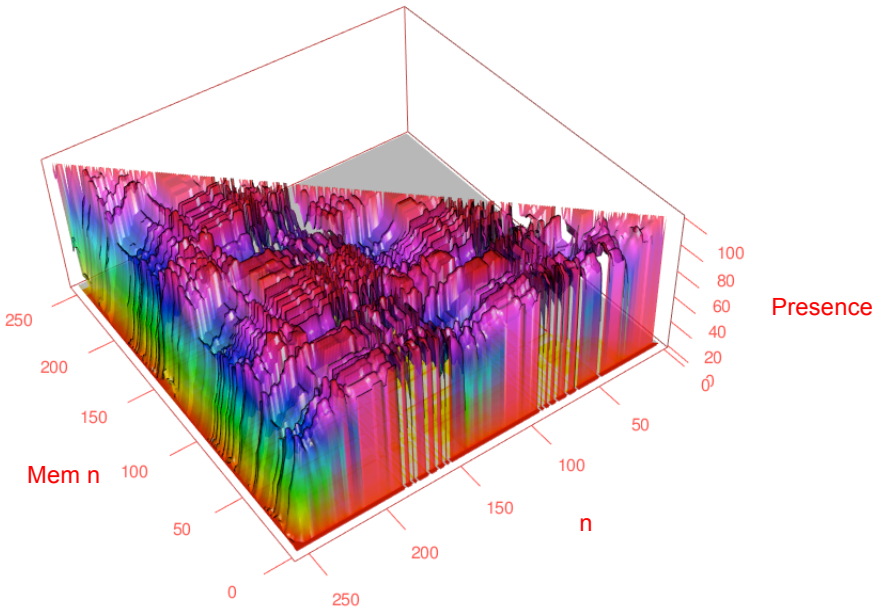

Fig. 51. Evolución de la memoria con el patrón 5 (exterior)

Patrón 6: *tres desconexiones en la frontera* ($P_F \rightarrow P_D \rightarrow P_F \rightarrow P_D \rightarrow P_F \rightarrow P_D \rightarrow P_F$)

Este patrón es similar al anterior salvo que se sitúa sólo cerca de la frontera. Su descomposición es la misma finalizando en P_F.

La Fig. 52 es similar a la Fig. 50. Se aprecia una pérdida de confianza más acusada, al encontrarse siempre al límite de la frontera. No da tiempo a recuperarla. Hay menos inicios de *buffering* porque las estimaciones son más negativas, al tener menos confianza corre menos riesgo. También en este caso la primera desconexión no se consigue predecir a tiempo pero sí las siguientes y en el intervalo en el que se sitúa el punto P_X.

La Fig. 53 tiene la misma forma que la Fig. 51. Es mucho más plana porque la situación es siempre crítica en la frontera, no variando acusadamente en el tiempo.

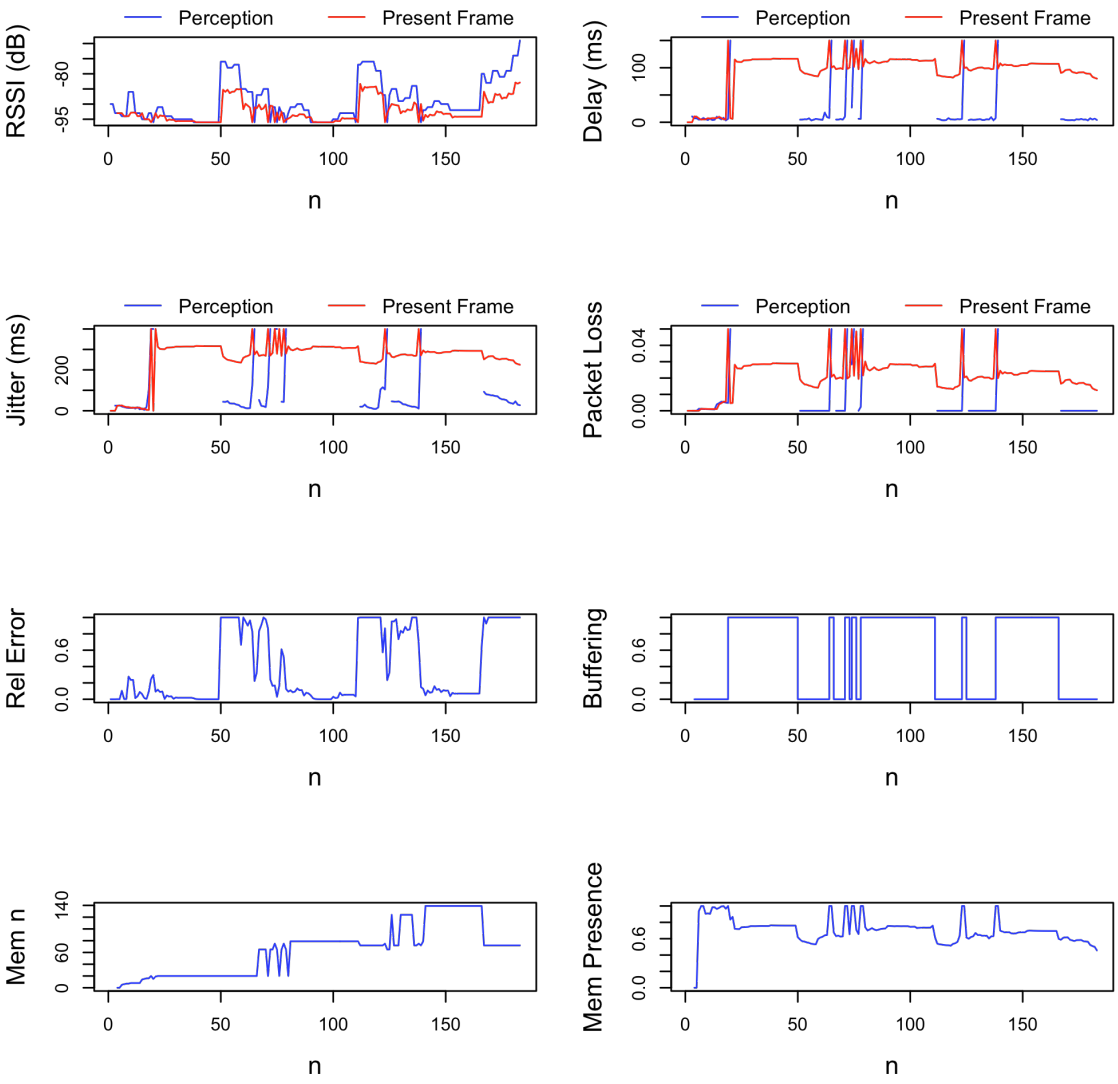

Fig. 52. Parámetros de rendimiento del patrón 6 (exterior)

Como se puede apreciar, los valores de todos los parámetros de QoS que se miden son coherentes y las estimaciones son adecuadas, lo cual indica que la probabilidad de asegurar la QoE es alta, por lo menos en un entorno exterior. Para un entorno interior repetimos las mismas pruebas resaltando las diferencias encontradas.

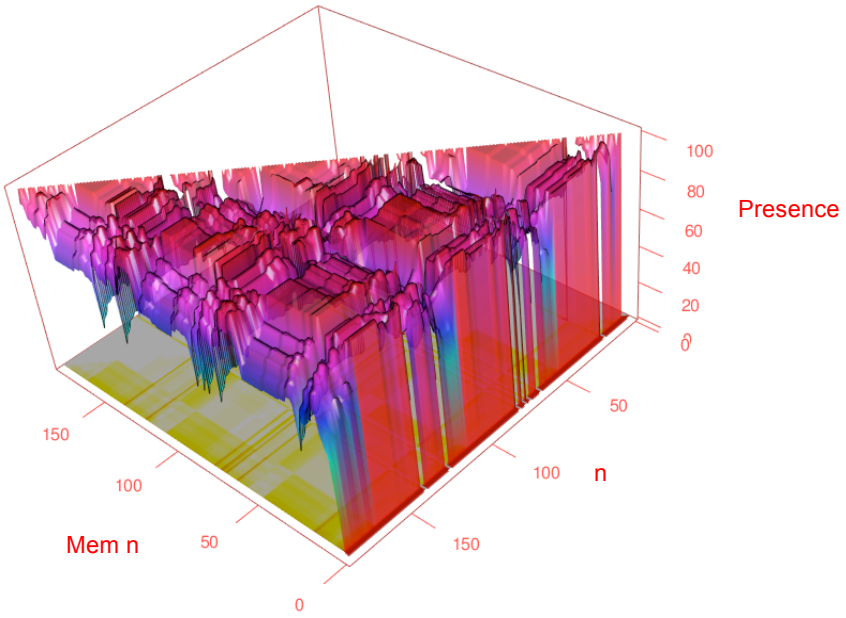

Fig. 53. Evolución de la memoria con el patrón 6 (exterior)

El rendimiento del patrón 1 (Fig. 54) y del patrón 2 (Fig. 55) en interior es prácticamente el mismo que en exterior. Sólo se observa en el jitter del patrón 1 un desfase mayor pero finalmente se ajusta correctamente. Las gráficas de evolución de la memoria con el patrón 1 (Fig. 56) y con el patrón 2 (Fig. 57) también tienen la misma forma y colores.

El rendimiento del patrón 3 (Fig. 58) y del patrón 4 (Fig. 59) en interior es muy parecido al de exterior salvo en la reconexión. En los dos casos se hacen varios intentos antes de poder reconectar con cierta estabilidad. Esto es debido a la situación fluctuante que se da en ese momento tal y como se observa en el RSSI. Las gráficas de evolución de la memoria con el patrón 3 (Fig. 60) y con el patrón 4 (Fig. 61) también son similares a las de exterior salvo por los intentos de reconexión.

El rendimiento del patrón 5 (Fig. 62) y del patrón 6 (Fig. 63) en interior también es análogo al de exterior. La primera desconexión no se consigue predecir a tiempo pero sí las posteriores en el intervalo en el que se sitúa el punto P_X. Las gráficas de evolución de la memoria con el patrón 5 (Fig. 64) y con el patrón 6 (Fig. 65) guardan una similitud destacable con sus homólogas de exterior.

Los resultados de las pruebas en interior son semejantes a los de las pruebas en exterior. No se aprecian diferencias notables más allá de hechos puntuales. Por lo tanto, en interior también es válido el método desarrollado.

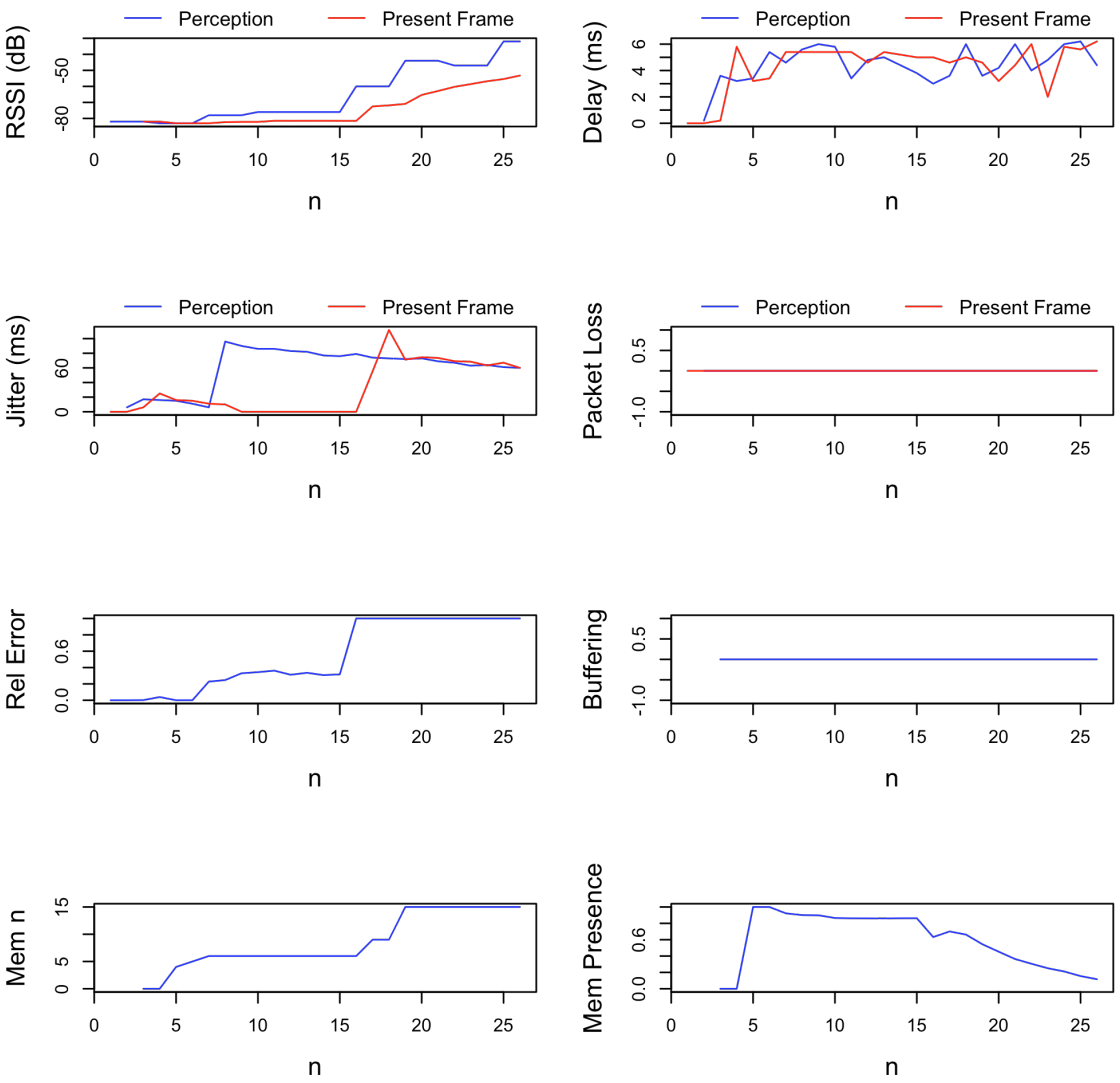

Fig. 54. Parámetros de rendimiento del patrón 1 (interior)

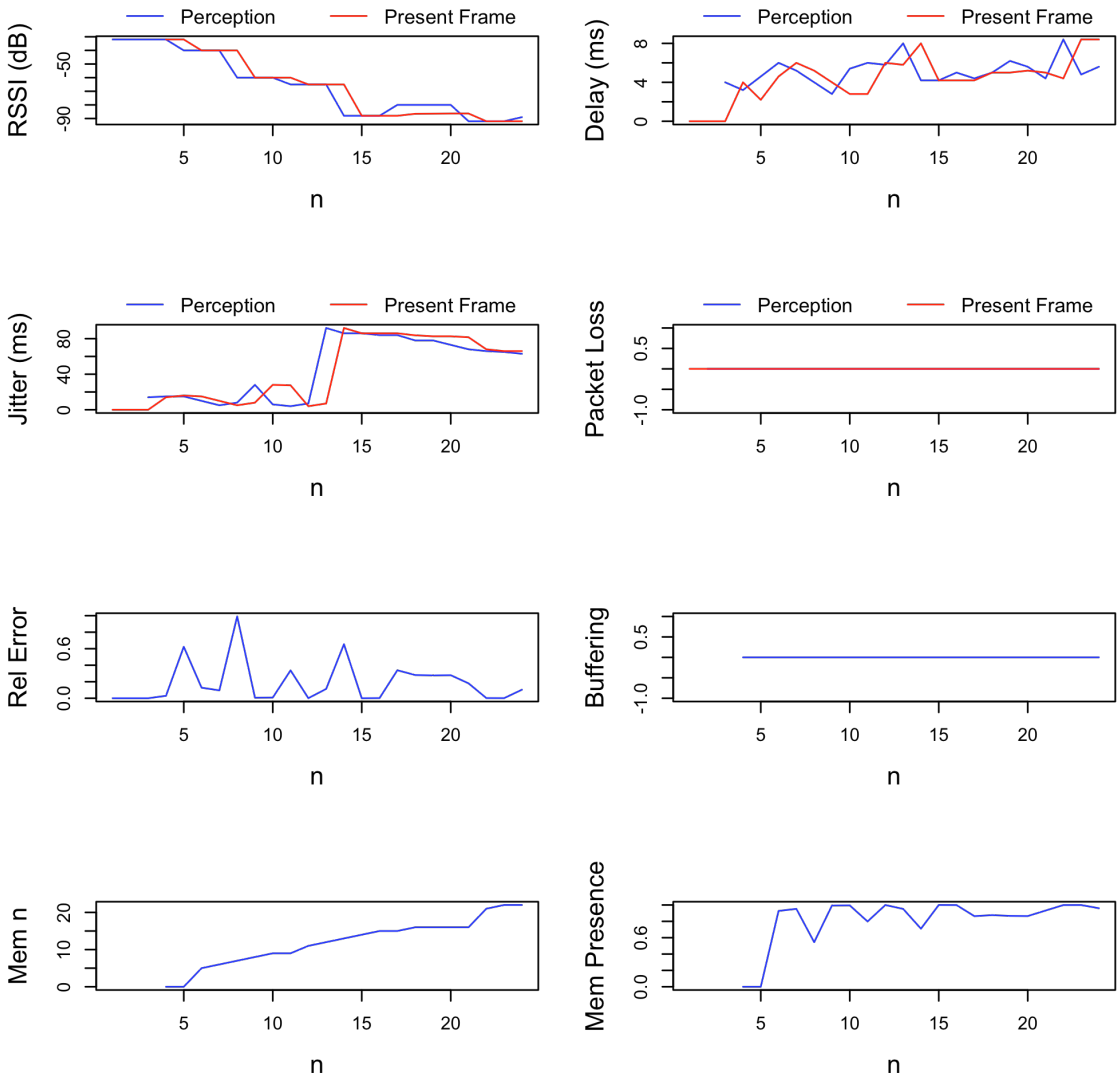

Fig. 55. Parámetros de rendimiento del patrón 2 (interior)

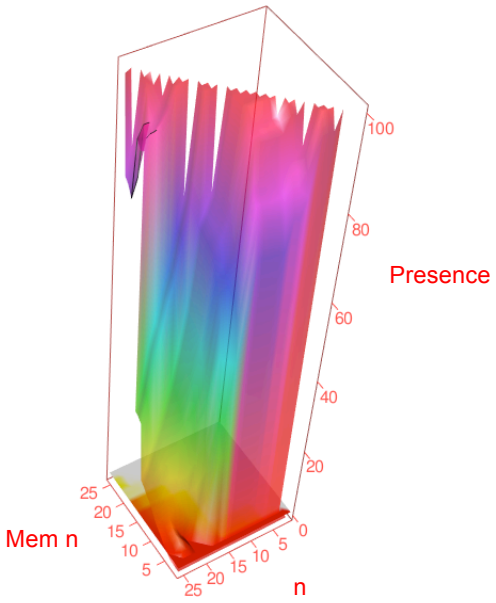

Fig. 56. Evolución de la memoria con el patrón 1 (interior)

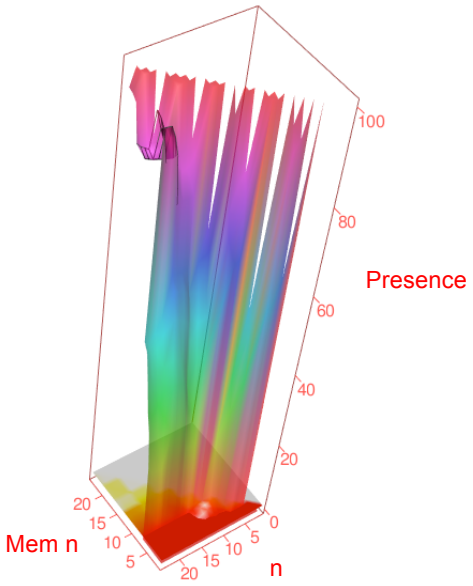

Fig. 57. Evolución de la memoria con el patrón 2 (interior)

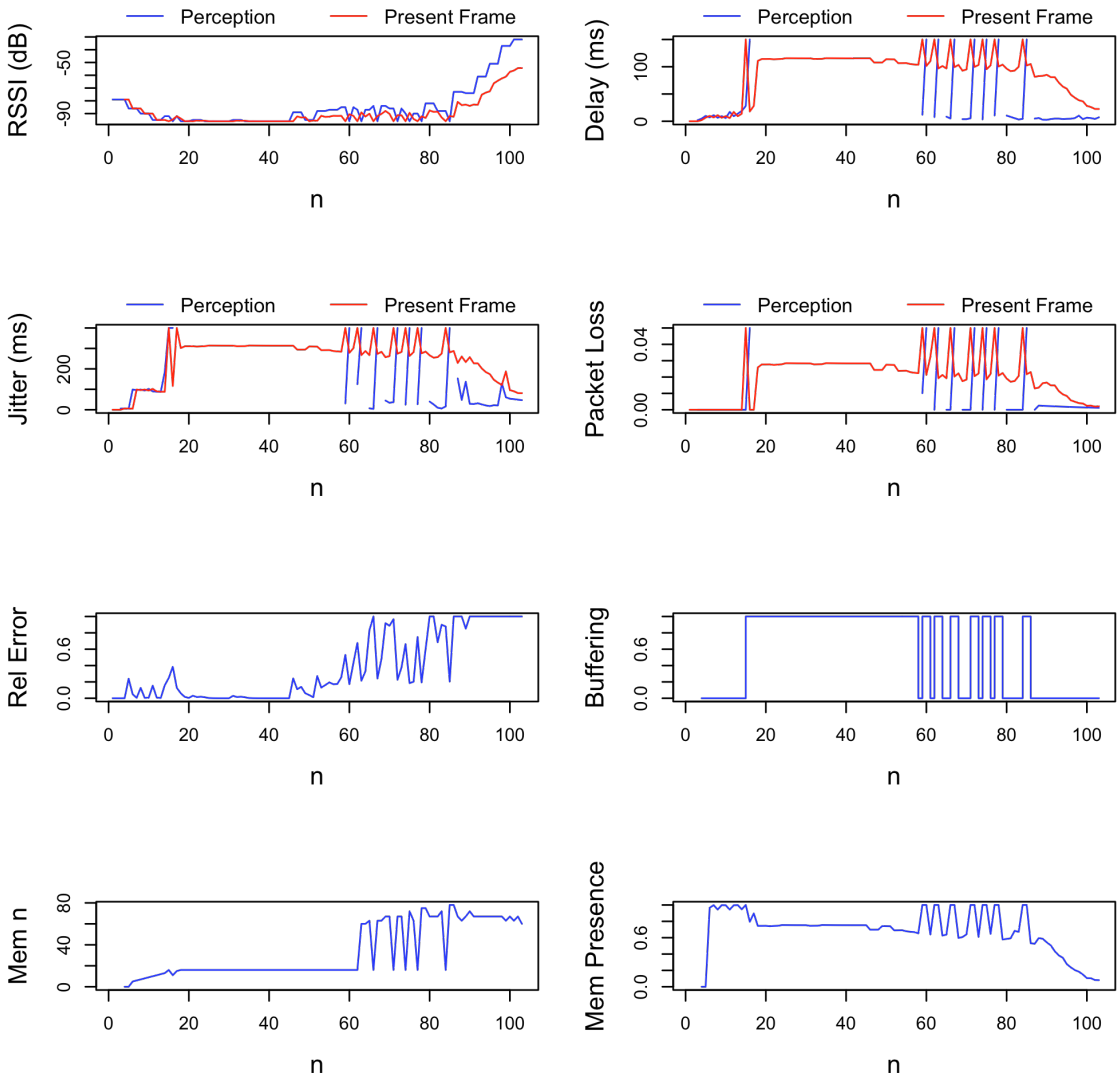

Fig. 58. Parámetros de rendimiento del patrón 3 (interior)

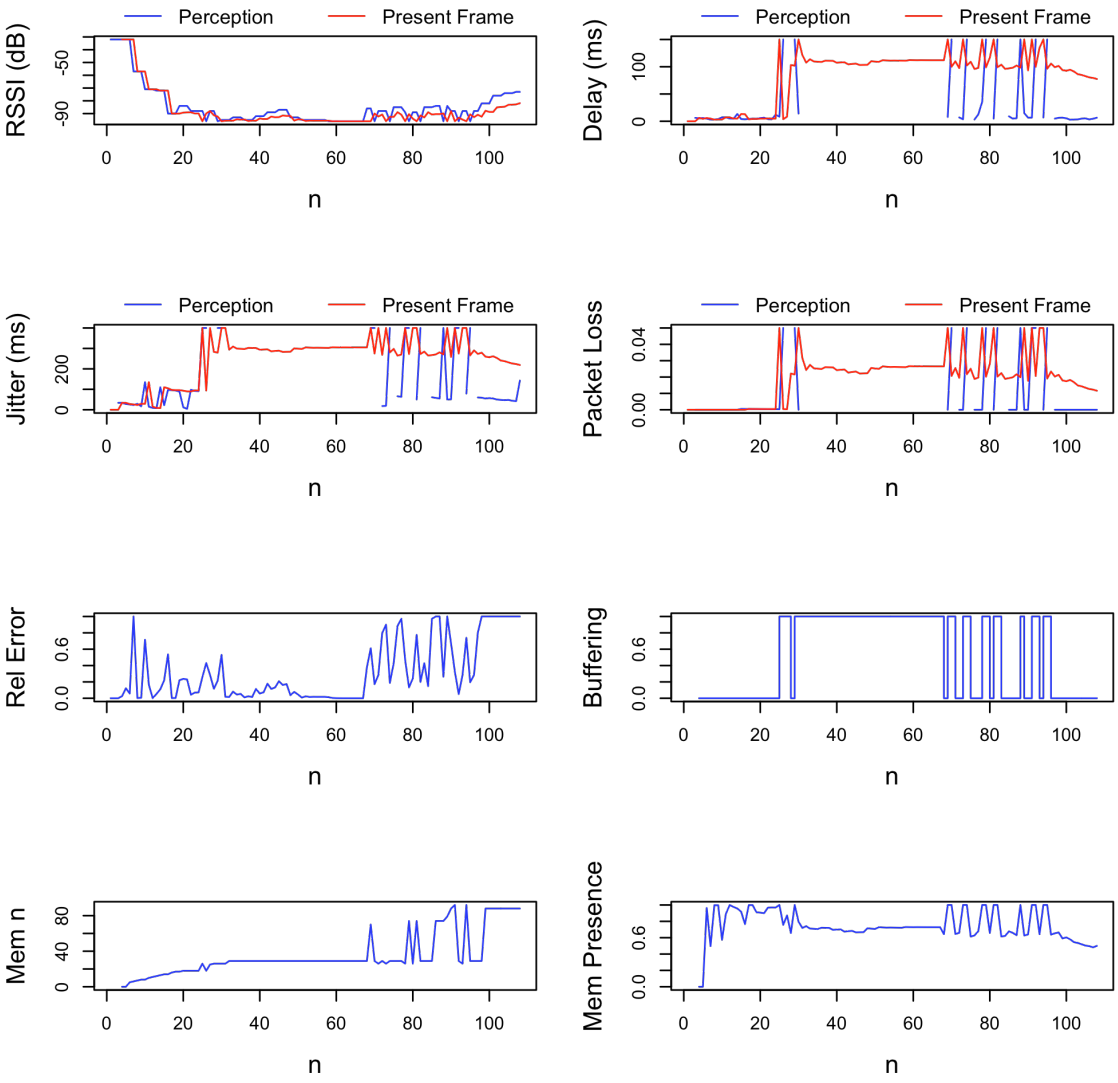

Fig. 59. Parámetros de rendimiento del patrón 4 (interior)

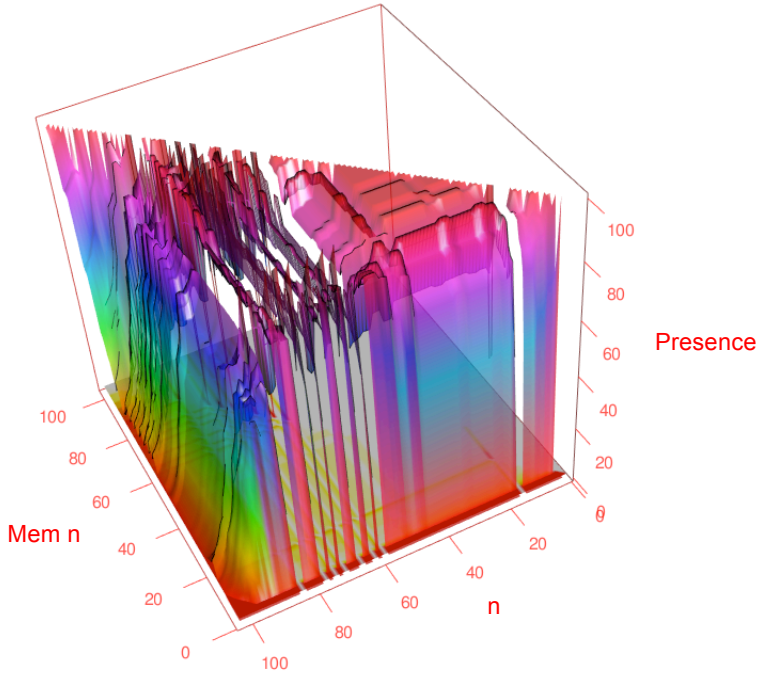

Fig. 60. Evolución de la memoria con el patrón 3 (interior)

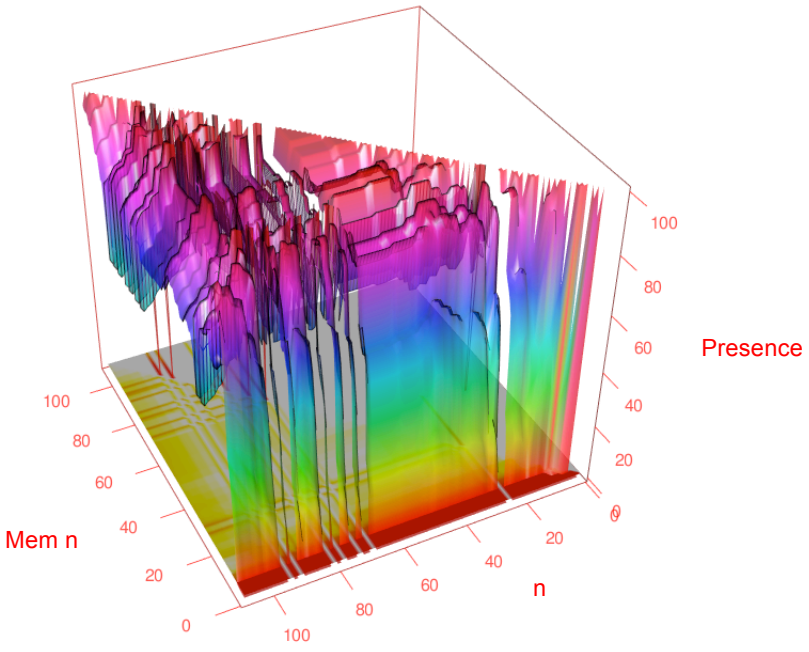

Fig. 61. Evolución de la memoria con el patrón 4 (interior)

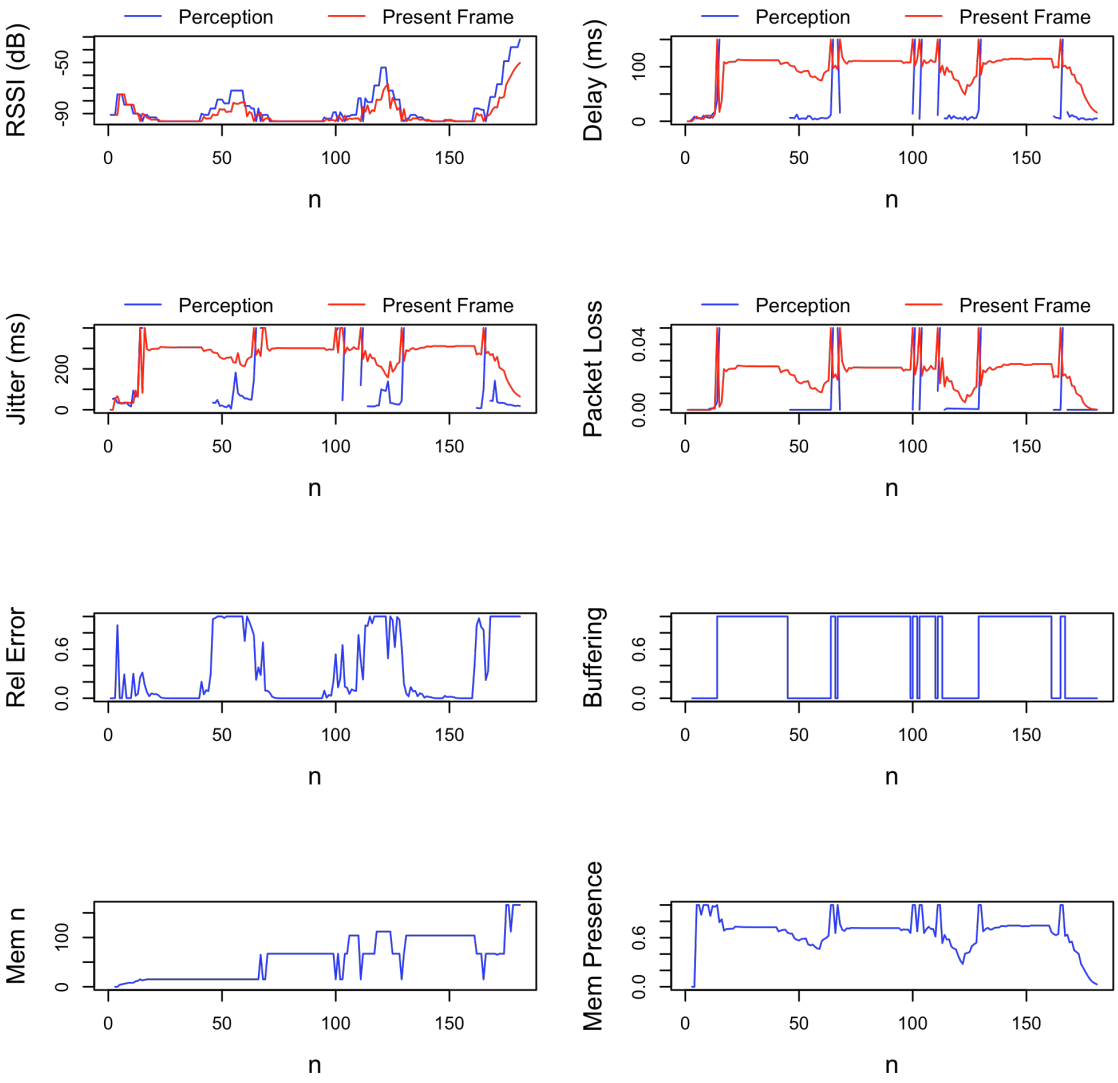

Fig. 62. Parámetros de rendimiento del patrón 5 (interior)

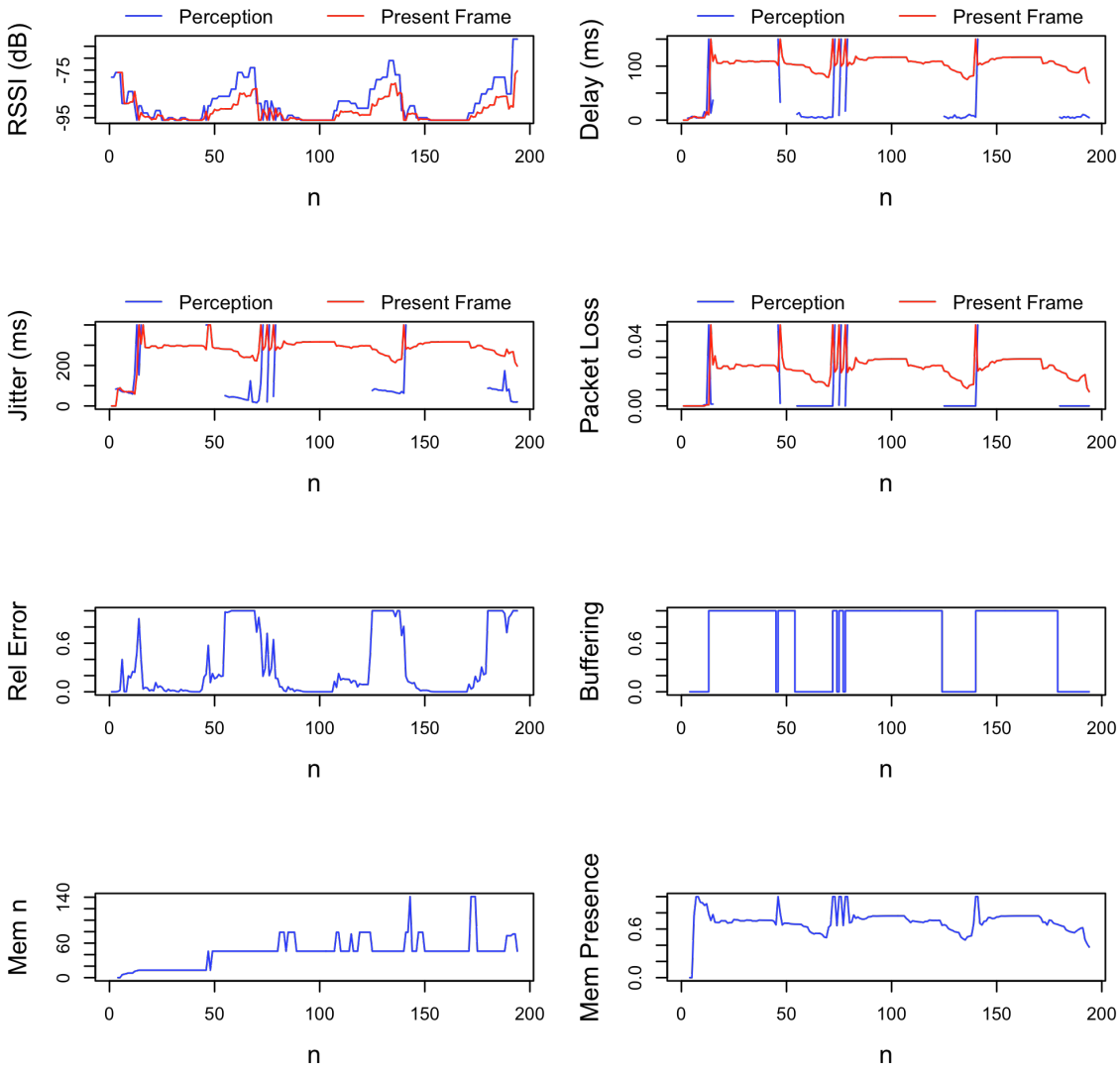

Fig. 63. Parámetros de rendimiento del patrón 6 (interior)

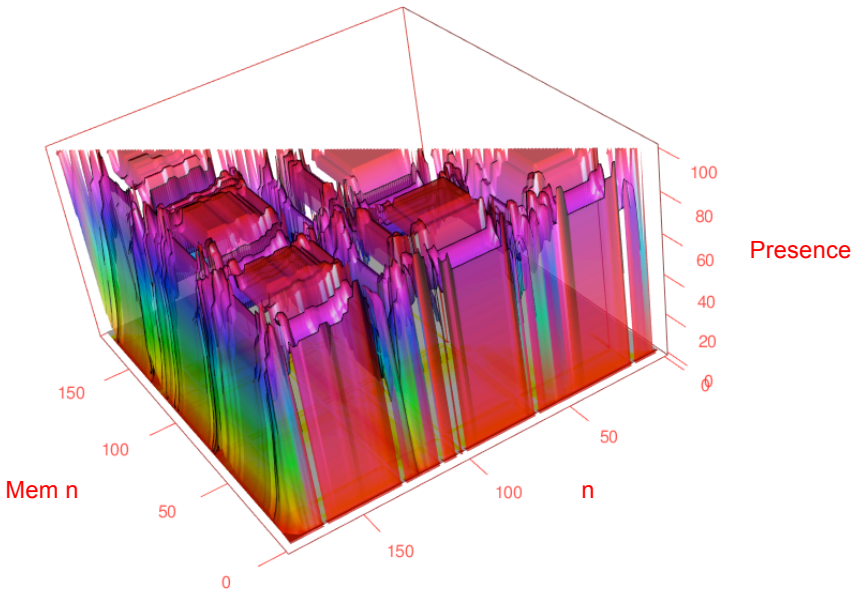

Fig. 64. Evolución de la memoria con el patrón 5 (interior)

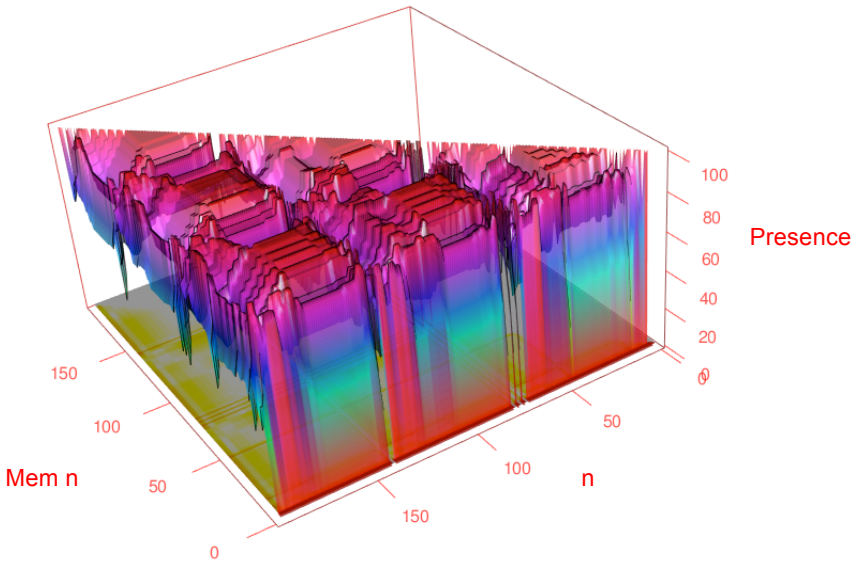

Fig. 65. Evolución de la memoria con el patrón 6 (interior)

4.5 Discusión

A partir de los resultados que hemos obtenido en las pruebas anteriores podemos concluir que los canales WiFi son caóticos deterministas porque sigue ciertos comportamientos reconocibles que son aprehendidos por nuestro método de predicción basado en emociones y agentes software emocionales.

Los resultados obtenidos en las pruebas dependen de los valores de los parámetros utilizados, los cuales determinan la naturaleza y la personalidad del agente emocional. Éstos son fruto de sucesivos ajustes en los que progresivamente se han ido mejorando los resultados obtenidos. Evidentemente se pueden seguir ajustando pero también es conveniente mostrar que no es indispensable disponer de valores cuasi-óptimos para operar con cierta eficacia. Aunque los resultados con el archivo de configuración de la Fig. 41 han sido satisfactorios en diferentes entornos, interiores y exteriores, y con diferentes dispositivos esto no garantiza que en otros entornos más concretos, como por ejemplo un sótano o la orilla de una playa, no haga falta un ajuste más fino. Esto quiere decir que nosotros hemos diseñado un método válido en muchos entornos y equipos diferentes pero que debe ser particularizado para entornos atípicos o poco habituales.

En las pruebas el error medio cometido es elevado, pero es menor en situaciones críticas (previas a una desconexión) y mayor en situaciones seguras. Esto se debe al carácter conservador que se le ha dado al método en este caso. Una vez experimentada una situación muy negativa necesita un tiempo para volver a coger confianza, es decir, para reducir su sesgo negativo en las predicciones. A pesar de este sesgo el error cometido en las predicciones es bajo cuando se observa a nivel físico.

Partiendo sin conocimiento alguno los agentes software por norma general no consiguen predecir la primera desconexión. Las siguientes sí lo hacen con una elevada tasa de acierto. En los casos en los que no lo consiguen, y se rebasa algún umbral de QoD, los agentes son capaces de reaccionar inmediatamente y sin demora minimizando la cantidad de información perdida si se termina produciendo la consecuente interrupción.

Finalmente se observa que es totalmente factible utilizar la arquitectura del proxy de este trabajo porque no añade retrasos inaceptables al iniciar la reproducción del video y éste se puede visualizar sin ningún tipo de alteración. Además la carga computacional añadida por el proxy-cliente en el dispositivo móvil es de muy baja y por tanto no afecta a la calidad del vídeo. La carga computacional del proxy-server también es baja, aunque gran parte de su actividad consista en operaciones en coma flotante. No obstante, todavía puede reducirse con computación paralela en coma flotante con tecnologías como *Open Computing Language* (*OpenCL*) [342].

En cuanto al sistema emocional desarrollado se observa que está justificado seguir trabajándo en él, más como método de aprendizaje por refuerzo, como los de Google DeepMind [343], que de razonamiento automático, como el IBM Watson [344]. También se ve que es posible establecer y estudiar una relación entre el marco presente del modelo y la conciencia, a la cual recientemente se le ha atribuido un carácter pasivo [345].

Capítulo 5

Conclusiones y líneas futuras

Resumen: para terminar se enumeran las conclusiones alcanzadas. También se proponen algunas líneas futuras de trabajo fundamentadas en los resultados obtenidos y el conocimiento aportado.

5.1 Conclusiones

El *video-streaming* es una técnica simple en teoría pero muy compleja en la práctica. La cantidad de requisitos y limitaciones que tiene asociada hace que su eficacia sea difícil de conseguir, y muchísimo más su eficiencia. Su uso en redes inalámbricas, las cuales se comportan como un sistema caótico determinista, es otra dificultad añadida. Hasta el momento las soluciones propuestas para este caso han sido principalmente de fuerza bruta, como emplear TCP, pero los resultados son decepcionantes. En este trabajo se ha planteado una alternativa a esta situación.

Nos hemos centrado en la mejora de la QoD sin necesidad de modificar los clientes y servidores de *video-streaming* existentes en la actualidad. Para ello hemos propuesto, diseñado y desarrollado un nuevo componente: un proxy dividido en dos partes implementado con agentes software emocionales que cubren la red inalámbrica y realizan una monitorización proactiva de su comportamiento. Este proxy evita que una desconexión suponga la pérdida de información o la anulación

de la sesión de *video-streaming* ahorrando en consecuencia molestias, tiempo y recursos computacionales, energéticos o de Red.

A pesar de que una red inalámbrica es un sistema caótico determinista y que normalmente en este tipo de entorno los agentes software sólo pueden comportarse reactivamente, hemos introducido un sistema que les hace capaces de realizar predicciones del estado del *video-streaming* con alto grado de acierto y de este modo tomar mejores decisiones y comportarse proactivamente.

Este sistema planteado para la predicción de sistemas caóticos deterministas está fundamentado en un modelo basado en emociones y que representa la combinación intuición-pensamiento. Este modelo se sitúa entre los niveles neuronal y psicológico, zona desierta en cuanto a conocimiento pero plagada de disputas y enfrentamientos como las relaciones entre cuerpo-mente, razón-emoción, conexionismo-cognitivismo, consciencia-inconsciente y así un largo etcétera. Estableciendo un símil entre animal y máquina, en el que las neuronas se corresponden con transistores y los comportamientos con los resultados, este modelo biológico se corresponde en cierta medida con lo que es el modelo de Von Neumann, el cual sigue la inmensa mayoría de computadores modernos.

Citas como “Mala memoria, la que sólo funciona hacia atrás” u “Ordenaré mis actos para que el presente sea toda la vida y les parezca el recuerdo” cobran un especial sentido con el contraintuitivo proceder de la intuición y el pensamiento, el cual es recogido en el sistema emocional que hemos propuesto. Aunque muy simplificado y sin pretender ser completamente realista se encuentra respaldado en gran parte por los estudios más recientes en neurociencia.

El sistema emocional de este trabajo se enmarca en el área de la inteligencia artificial en la que el objetivo de crear un sistema que piense como un ser humano

ha ido perdiendo fuerza paulatinamente por su dificultad y escasos resultados. Evidentemente resulta más complicado recrear un ser humano ideal, sólo con cualidades consideradas positivas, porque simplemente no existe. En el sistema emocional no se busca ni se evita ninguna cualidad en particular. La posibilidad de enfermedades mentales virtuales o la reaparición de la malsana tendencia tan humana de mantener emociones durante períodos largos de tiempo como la tristeza, que puede llevar a la autodestrucción, el miedo-odio, que puede llevar a la agresión, o la frustración, que puede llevar al destrozo del entorno, son una posibilidad y un riesgo a tener en cuenta. A cambio se consigue un sistema más coherente y realista.

De momento el sistema emocional introducido en este trabajo es una aportación viable para seguir avanzando en la emulación de la combinación intuición-pensamiento. Está en consonancia con las teorías más recientes en neurociencia de las que tenemos conocimiento. Proporciona nuevas posibilidades interesantes con agentes software para desarrollar diferentes tipos de comportamientos. A largo plazo alcanzar seres humanos virtuales considerados equilibrados, como un Gregor Samsa o un Übermensch, no se contempla, pero sí agentes software emocionales autónomos capaces de ubicarse en entornos caóticos deterministas, siendo ejemplos de vida artificial sintética.

5.2 Líneas futuras de trabajo

“Toda historia es una Historia Interminable” y ésta no es menos. Por ello como trabajos futuros proponemos:

- *Soporte de más protocolos de streaming, incluidos los basados en HTTP.* Presumiblemente para estos últimos el proxy desarrollado reduce la congestión de la red inalámbrica al emplear UDP en sus comunicaciones internas, entre proxy-cliente y proxy-servidor, independientemente del protocolo concreto usado por el cliente y el servidor de *streaming*.

- *Exploración de versiones más biológicas de los métodos estudiados.* Redes neuronales artificiales con células gliales, algoritmos genéticos no dirigidos por una función de aptitud sino por una de suerte, y sistemas de lógica difusa con términos lingüísticos dinámicos y adaptativos.

- *Formulación de mecanismos para que agentes software emocionales puedan compartir conocimiento efectivamente.* Por ejemplo, experiencias comunes en base a geolocalización.

- *Continuar y ampliar el diseño del sistema emocional introducido en este trabajo.* Incorporar la capacidad de realizar acciones. Aplicación en otros problemas que impliquen sistemas caóticos deterministas.

Sin duda el "especialismo" denunciado por Ortega y Gasset se ve como la mayor amenaza para realizar y continuar este trabajo porque para avanzar en la ciencia y tecnología actual ya no basta sólo con equipos interdisciplinares. La interdisciplinariedad debe darse sobre todo a nivel personal; la composición de una obra orquestal requiere de más que de cierto conocimiento de todos los instrumentos implicados en ella.

Bibliografía

[1] Real Academia Española, "Diccionario de la lengua española. 22ª edición", Espasa Calpe, Madrid, 2001.

[2] T. Mikkonen, "Programming Mobile Devices: An Introduction for Practitioners", Wiley, Chichester, 2007.

[3] S. Bicheno, "Global Smartphone Installed Base Forecast by Operating System for 88 Countries: 2007 to 2017", Strategy Analytics Wireless Smartphone Strategies (WSS), 2012.

[4] D.E. Comer, "The Internet Book: Everything You Need to Know About Computer Networking and How Internet Works, 4th Edition", Prentice Hall, 2006.

[5] "Global Positioning System (GPS) Standard Positioning Service (SPS) Performance Standard, 4th ed.", Department of Defense United States of America - Navstar Global Positioning System, septiembre 2008.

[6] W. Simpson, "Video Over IP: IPTV, Internet Video, H.264, P2P, Web TV, and Streaming: A Complete Guide to Understanding the Technology (2nd Edition)", Focal Press, 2008.

[7] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2013-2018", Cisco White Paper, 2014.

[8] "IEEE Std 802.11-2012: IEEE Standard for Information technology – Telecommunications and information exchange between systems - Local and metropolitan area networks – Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications", IEEE Computer Society, marzo 2012.

[9] H. Holma, A. Toskala, "HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobile Communications", Wiley, 2006.

[10] E. Dahlman, S. Parkvall, J. Skold, "4G: LTE/LTE-Advanced for Mobile Broadband", Academic Press, 2011.

[11] D. Tse, P. Viswanath, "Fundamentas of Wireless Communication", Cambridge University Press, 2005.

[12] S.H. Strogatz, "Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry, and Engineering", Westview Press, 2001.

[13] H. Kantz, T. Schreiber, "Nonlinear Time Series Analysis, 2nd edition", Cambridge University Press, 2004.

[14] C.M. Bishop, "Pattern Recognition and Machine Learning", Springer, New York, 2006.

[15] E.R. Kandel, J.H. Schwartz, T.M. Jessell, S.A. Siegelbaum, A.J. Hudspeth (Eds.), "Principles of Neural Science, 5th Edition", McGraw-Hill, New York, 2012.

[16] M. Lewis, J.M. Haviland-Jones, L.F. Barrett (Eds.), "Handbook of Emotions, 3rd Edition", The Guilford Press, New York, 2010.

[17] B. Vandalore, W-C. Feng, R. Jain, S. Fahmy, "A Survey of Application Layer Techniques for Adaptive Streaming of Multimedia", Real-Time Imaging, Vol. 7, No. 3, pp. 221-235, 2001.

[18] J. Chakareski, P. A. Chou, "RaDiO edge: Rate-distortion Optimized Proxy-Driven Streaming from the Network Edge", IEEE/ACM Transactions on Networking, pp. 1302 – 1312, diciembre 2006.

[19] L. Gao, Z-L. Zhang, D. Towsley, "Proxy-Assisted Techniques for Delivering Continuous Multimedia Streams", IEEE/ACM Transactions on Networking, Vol. 11, No. 6, pp. 884-894, diciembre 2003.

[20] P. Bellavista, A. Corradi, C. Giannelli, "Mobile Proxies for Proactive Buffering in Wireless Internet Multimedia Streaming", Proceedings of the IEEE International Conference on Distributed Computing Systems Workshop (ICDCSW'05), pp. 297-304, junio 2005.

[21] P. Bala Krishna Prasad, G. Murali, G. Gurukesava Das, K. Siva Krishna Rao, "Congestion Controlling for Streaming Media Trough Buffer Management and Jitter Control", International Journal of Computer Science and Network Security (IJCSNS), Vol. 9, No. 2, pp. 362-371, 2009.

[22] A. Dua, N. Bambos, "Buffer Management for Wireless Media Streaming", In Proceedings of the IEEE Global Telecommunications Conference 2007 (GLOBECOM '07), pp. 5226-5230, 2007.

[23] J. Ridenour, J. Hu, N. Pettis, Y-H. Lu, "Low-Power Buffer Management for Streaming Data", IEEE Transactions on Circuits and Systems for Video Technology, Vol. 17, No. 2, pp. 143-157, 2007.

[24] Y-H. Zhang, K.H. Li, C.-Q. Xu, L-M. Sun, "Joint Rate Allocation and Buffer Management for Robust Transmission of VBR Video", Acta Automatica Sinica, Vol. 34, No. 3, pp. 337-343, 2008.

[25] G. Liang, B. Liang, “Effect of Delay and Buffering on Jitter-Free Streaming Over Random VBR Channels”, IEEE Transactions on Multimedia, Vol. 10, No. 6, pp. 1128-1141, 2008.

[26] G. Scalosub, P. Marbach, J. Liebeherr, “Buffer Management for Aggregated Streaming Data with Packet Dependencies”, In Proceedings IEEE INFOCOM 2010, pp. 1-5, 2010.

[27] H.B. Kazemian, “An Intelligent Video Streaming Technique in Wireless Communications”, In Proceedings of the 3rd International Forum on Applied Wearable Computing (IFAWC 2006), art. 23, 2006.

[28] Z. Orlov, “Network-driven adaptive video streaming in wireless environments”, In Proceedings of the IEEE 19th International Symposium on Personal, Indoor and Mobile Radio Communications, 2008 (PIMRC 2008), pp. 1-6, 2008.

[29] S-R. Tong, S.-H. Yang, “Buffer Control to Support a Seamless Stream Handoff in a WLAN that Employs Simulcast Streaming”, IEEE Transactions on Wireless Communications, Vol. 7, No. 1, pp. 260-268, 2008.

[30] T. Lin, C. Wang, P. Lin, “A Neural Network Based Context-Aware Handoff Algorithm for Multimedia Computing”, ACM Transactions on Multimedia Computing, Communications, and Applications, Vol. 4, No. 3, 2008.

[31] K. Atalah, E.M. Macías, A. Suárez, “A Proactive Horizontal Handover Algorithm Based on RSSI Supported by a New Gradient Predictor”, UbiCC Journal, Vol. 3, No. 3, 2008.

[32] A.G. Fragkiadakis, E.Z. Tragos, I.G. Askoxylakis, “Video Streaming Performance in Wireless Hostile Environments”, In: MUE ’11: Proceedings of the 2011 Fifth FTRA International Conference on Multimedia and Ubiquitous Engineering, pp. 267-272, Loutraki, Grecia, 2011.

[33] P. Lin, T. Lin, C. Wang, “Performance Analysis of a Cross-Layer Handoff Ordering Scheme in Wireless Networks”, IEEE Transactions on Wireless Communications, Vol. 7, No. 12, pp. 5166-5171, 2008.

[34] C. Bouras, A. Gkamas and G. Kioumourtzis, “Challenges in Cross Layer Design for Multimedia Transmission over Wireless Networks”, Wireless World Research Forum (WWRF21), Stockholm, Working Group 3, 2008.

[35] P-H. Hsiao, H.T. Kung, K.S. Tan, “Streaming Video over TCP with Receiver-based Delay Control”, In: NOSSDAV '01: Proceedings of the 11th international workshop on Network and operating systems support for digital audio and video, pp. 199-208, New York, 2001.

[36] S. Schütz, M. Harvan, L. Eggert, S. Schmid, M. Brunner, “Protocol Enhancements for Disruption Tolerant Mobile Networking”, In the 25th IEEE Conference on Computer Communications (INFOCOM 2006), 2006.

[37] T. Weingart, D.C. Sicker, D. Grunwald, "Identifying Opportunities for Exploiting Cross-Layer Interactions in Adaptive Wireless Systems", In H. Wang, J. Huang, M. Van Der Schaar, D.O. Wu, Z. Han (Eds.) Cross-Layer Optimised Wireless Multimedia Communications, Hindawi Publishing Corporation, Art. 49604, Cairo, Egypt, 2007.

[38] M. van der Schaar, D.S. Turaga, "Cross-Layer Packetization and Retransmission Strategies for Delay-Sensitive Wireless Multimedia Transmission", IEEE Transactions on Multimedia, Vol. 9, No. 1, pp. 185-197, 2007.

[39] S. Jiang, Y. Xue, D. Schmidt, "Disruption-Aware Service Composition and Recovery in Dynamic Networking Environments", In Workshops on Automating Service Quality, 2007 (WRASQ 2007), Atlanta, 2007.

[40] M. Caleffi, L. Paura, "Opportunistic Routing for Disruption Tolerant Networks", In Proceedings of the 2009 International Conference on Advanced Information Networking and Applications Workshops (WAINA '09), pp. 826-831, 2009.

[41] B. Burns, O. Brock, B.N. Levine, "Autonomous enhancement of disruption tolerant networks", In Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA 2006), pp. 2105-2110, 2006.

[42] S-Y. Wu, J. Hsu, C.-M. Chen, "Headlight Prefetching and Dynamic Chaining for Cooperative Media Streaming in Mobile Environments", IEEE Transactions on Mobile Computing, Vol. 8, No. 2, pp. 173-187, 2009.

[43] N. Baghaei, R. Hunt, "Quality of Service Support in Mobile Multimedia Networks", In M. Ma, M.K. Denko, Y. Zhang (Eds.) Wireless Quality of Service: Techniques, Standards, and Apllications, pp. 1-25, Auerbach Publications, Boca Raton, USA, 2009.

[44] R. Goel, M. Sarkar, "Enhancing QoS for Streaming Video over WLANs", In Proceedings of the World Congress on Engineering and Computer Science 2008 (WCECS 2008), pp. 76-85, 2008.

[45] A. Kumar B.R., L.C. Reddy, P.S. Hiremath, Naresh S.S., "RTSP Audio and Video Streaming for QoS in Wireless Mobile Devices", IJCSNS International Journal of Computer Science and Network Security, Vol. 8, No. 1, pp. 96-101, 2008.

[46] E.M. Macías, A. Suárez, "Estimación proactiva de la calidad de recepción de vídeo streaming en WiFi usando una técnica de cross-layer", IEEE Latin America Transactions, Vol. 7, No. 3, pp. 383-389, 2009.

[47] J. Hu, S. Choudhury, J.D. Gibson, "Video Capacity of WLANs with a Multiuser Perceptual Quality Constraint", IEEE Transactions on Multimedia, Vol. 10, No. 8, pp. 1465-1478, 2008.

[48] L. Janowski, Z. Papir, "Modeling subjetive tests of quality of experience with a Generalizad Linear Model", In Proceedings of the 2009 Intenational Workshop on Quality of Multimedia Experience (QoMEx 2009), pp. 35-40, 2009.

[49] H. Batteram, G. Damm, A. Mukhopadhyay, L. Philippart, R. Odysseos, C. Urrutia-Valdés, “Delivering Quality of Experience in Multimedia Networks”, Bell Labs Technical Journal, Vol. 15, No. 1, pp. 175-194, 2010.

[50] K. Piamrat, C. Viho, A. Ksentini, J-M. Bonnin, “Quality of Experience Measurements for Video Streaming over Wireless Networks”, Proceedings of the 2009 Sixth International Conference on Information Technology: New Generations (ITNG '09), pp. 1184-1189, Las Vegas, Nevada, 2009.

[51] S.S. Krishnan, R.K. Sitaraman, “Video Stream Quality Impacts Viewer Behavior: Inferring Causality Using Quasi-Experimental Designs”, IEEE/ACM Transactions on Networking, Vol. 21, No. 6, pp. 2001-2014, 2013.

[52] R.K.P. Mok, E.W.W. Chan, R.K.C. Chang, “Improving TCP Video Streaming QoE by Network QoS Management”, 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) (Mini-conference), 2011.

[53] N. Cranley, G-M. Muntean, “Improving User-Perceived Quality for Video Streaming over WLAN”, In Y. Zhang, S. Mao, L.T. Yang, T.M. Chen (Eds.) Broadband Mobile Multimedia: Techniques and Applications, pp. 361-406, Auerbach Publications, Boca Raton, USA, 2008.

[54] M. Mu, E. Cerqueira, F. Boavida, A. Mauthe, “Quality of Experience Management Framework for real-time multimedia applications”, International Journal of Internet Protocol Technology (IJIPT), Vol. 4, No. 1, pp. 54-64, 2009.

[55] D. Rodrigues, E. Cerqueira, E. Monteiro, “Quality of Service and Quality of Experience in Video Streaming”, In Proceedings of the International Workshop on Traffic Management and Traffic Engineering for the Future Internet (FITraMEn2008), EuroNF NoE, 2008.

[56] E.S. Aguiar, B.A. Pinheiro, J.F.S. Figueirêdo, E. Cerqueira, A.J.G. Abelém, R.L. Gomes, “Trends and Challenges for Quality of Service and Quality of Experience for Wireless Mesh Networks”, In N. Funabiki (Ed.) Wireless Mesh Networks, pp. 127-148, InTech, Rijeka, Croatia, 2011.

[57] M. Bali, “Drools JBoss Rules 5.X Developer's Guide”, Packt Publishing, Birmingham, UK, 2013.

[58] P. Coulton, R. Edwards, H. Clemson, “S60 Programming: A Tutorial Guide (Symbian Press)”, John Wiley & Sons, Chichester, England, 2007.

[59] Nokia N95. Online: [http://www.microsoft.com/en-us/mobile/phone/nokia-n95/specifications/].

[60] Android Gingerbread. Online: [http://developer.android.com/about/versions/android-2.3-highlights.html].

[61] Samsung Galaxy Ace. Online: [http://www.gsmchoice.com/en/catalogue/samsung/gts5830ace/].

[62] Android Jelly Bean. Online: [http://developer.android.com/about/versions/jelly-bean.html].

[63] Samsung Galaxy Ace 2. Online: [http://www.gsmchoice.com/en/catalogue/samsung/galaxyace2/].

[64] Android KitKat. Online: [http://developer.android.com/about/versions/kitkat.html].

[65] Samsung Galaxy Ace 4. Online: [http://www.gsmchoice.com/en/catalogue/samsung/smg357fz/].

[66] Android Lollipop. Online: [http://developer.android.com/about/versions/lollipop.html].

[67] Motorola Moto G (3rd Gen.). Online: [http://www.gsmchoice.com/en/catalogue/motorola/motog2015/Motorola-Moto-G-2015.html].

[68] Microsoft Windows. Online: [http://www.microsoft.com/en-us/windows/].

[69] Ubuntu Linux. Online: [http://www.ubuntu.com/].

[70] Mac OS X. Online: [https://www.apple.com/osx/].

[71] B.W. Kernighan, D.M. Ritchie, "The C Programming Language, 2nd Edition", Prentice Hall, 1988.

[72] B. Stroustrup, "The C++ Programming Language, 4th Edition", Addison-Wesley, 2013.

[73] H. Schildt, "Java The Complete Reference, 9th Edition", McGraw-Hill, 2014.

[74] D.M. Beazley, "Python Essential Reference, 4th Edition", Addison-Wesley, 2009.

[75] A. Suárez, F. Espino, E.M. Macías, "Uso de JADE-LEAP para recuperar automáticamente sesiones de vídeo-streaming en teléfonos móviles", XVIII Jornadas TELECOM I+D, Bilbao, España, 2008.

[76] A. Suárez, F. Espino, E.M. Macías, "Automatic recovering of RTSP sessions in mobile telephones using JADE-LEAP", IEEE Latin America Transactions, Vol. 7, No. 3, pp. 410-417, 2009.

[77] E. Macías, A. Suárez, F. Espino, "Multi-platform Video Streaming Implementation on Mobile Terminals", In A. Suárez, E. Macías (Eds.) Multimedia Services and Streaming for Mobile Devices: Challenges and Innovations, pp. 288-314, Information Science Reference, Hershey, PA, 2012.

[78] P.A. Chou, "Streaming media on Demand and Live Broadcast", In P.A. Chou, M. Van der Schaar (Eds.) Multimedia over IP and Wireless Networks: Compression, Networking, and Systems, pp. 453-501, Academic Press, USA, 2007.

[79] MIDI Manufacturers Association, "Complete MIDI 1.0 Detailed Specification", MIDI Manufacturers Association Incorporated, 1999/2008.

[80] A.M. Bock, “Video Compression Systems”, The Institution of Engineering and Technology, London, United Kingdom, 2009.

[81] ISO/IEC 10918-1:1994, “Information technology -- Digital compression and coding of continuous-tone still images: Requirements and guidelines”, 1994.

[82] ITU-T Rec. H.261 (03/93), “Video codec for audiovisual services at p x 64 kbit/s”, 1993.

[83] ISO/IEC 11172-2:1993, “Information technology – Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 2: Video”, 1993.

[84] ISO/IEC 13818-2:2000, “Information technology – Generic coding of moving pictures and associated audio information: Video”, 2000.

[85] ITU-T Rec. H.263 (01/2005), “Video coding for low bit rate communication”, 2005.

[86] ISO/IEC 14496-2:2004, “Information technology - Coding of audio-visual objects - Part 2: Visual”, 2004.

[87] ISO/IEC 14496-10:2012, “Information technology - Coding of audio-visual objects - Part 10: Advanced Video Coding”, 2012.

[88] ITU-T Rec. H.265 (04/2013), “High efficiency video coding”, 2013.

[89] M. Handley, V. Jacobson, “SDP: Session Description Protocol”, IETF RFC 2327, 1998.

[90] H. Schulzrinne, A. Rao, R. Lanphier, “Real Time Streaming Protocol (RTSP)”, IETF RFC 2326, 1998.

[91] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, T. Berners-Lee, “Hypertext Transfer Protocol -- HTTP/1.1”, IETF RFC 2068, 1997.

[92] T. Berners-Lee, L. Masinter, M. McCahill, “Uniform resource locators (URL)”, IETF RFC 1738, 1994.

[93] J. Postel, “Internet Protocol”, IETF RFC 791, 1981.

[94] J. Postel, “Transmission Control Protocol”, IETF RFC 793, 1981.

[95] J. Postel, “User Datagram Protocol”, IETF RFC 768, 1980.

[96] H. Schulzrinne, S. Casner, R. Frederick, V. Jacobson, “RTP: A Transport Protocol for Real-Time Applications”, IETF RFC 3550, 2003.

[97] RealNetworks, “RDT Feature level 3.0 – Design Specification”, RealNetworks, Inc., 2005.

[98] Microsoft, “[MS-MMSP]: Microsoft Media Server (MMS) Protocol”, Microsoft Corporation, 2013.

[99] M. Thornburgh, “Adobe's Secure Real-Time Media Flow Protocol”, IETF, 2013.

[100] H. Schulzrinne, S. Casner, "RTP Profile for Audio and Video Conferences with Minimal Control", IETF RFC 3551, 2003.

[101] L. Gharai, C. Perkins, "RTP Payload Format for Uncompressed Video", IETF RFC 4175, 2005.

[102] L. Berc, W. Fenner, R. Frederick, S. McCanne, P. Stewart, "RTP Payload Format for JPEG-compressed Video", IETF RFC 2435, 1998.

[103] R. Even, "RTP Payload Format for H.261 Video Streams", IETF RFC 4587, 2006.

[104] D. Hoffman, G. Fernando, V. Goyal, M. Civanlar, "RTP Payload Format for MPEG1/MPEG2 Video", IETF RFC 2250, 1998.

[105] J. Ott, C. Bormann, G. Sullivan, S. Wenger, R. Even, "RTP Payload Format for ITU-T Rec. H.263 Video", IETF RFC 4629, 2007.

[106] M. Schmidt, F. de Bont, S. Doehla, J. Kim, "RTP Payload Format for MPEG-4 Audio/Visual Streams", IETF RFC 6416, 2011.

[107] Y-K. Wang, R. Even, T. Kristensen, R. Jesup, "RTP Payload Format for H.264 Video", IETF RFC 6184, 2011.

[108] T. Schierl, S. Wenger, Y-K. Wang, M.M. Hannuksela, Y. Sanchez, "RTP Payload Format for High Efficiency Video Coding", IETF Internet-Draft, 2013.

[109] B. Wang, J. Kurose, P. Shenoy, D. Towsley. "Multimedia streaming via TCP: An analytic performance study", Journal ACM Transactions on Multimedia Computing, Communications, and Applications (TOMCCAP), Vol. 4, No. 2, Art. 16, 2008.

[110] I. Sodagar, "The MPEG-DASH Standard for Multimedia Streaming Over the Internet", IEEE MultiMedia, pp. 62-67, 2011.

[111] T. Stockhammer, "Dynamic Adaptive Streaming over HTTP-Design Principles and Standards", In: MMSys '11: Proceedings of the 2nd annual ACM conference on Multimedia systems, pp. 133-144, New York, USA, 2011.

[112] Adobe Flash Platform, "HTTP Dynamic Streaming on the Adobe Flash Platform", Adobe Systems Incorporated, 2010.

[113] A. Zambelli, "IIS Smooth Streaming Technical Overview", Microsoft Corporation, 2009.

[114] R. Pantos, W. May, "HTTP Live Streaming", IETF, 2013.

[115] ISO/IEC 23009-1:2012, "Information technology – Dynamic adaptative streaming over HTTP (DASH) – Part 1: Media presentation description and segment formats".

[116] iOS Dev Center – Apple Developer. Online: [http://developer.apple.com/devcenter/ios/].

[117] Android Developers. Online: [http://developer.android.com/].

[118] H. Parmar, M. Thornburgh, "Adobe's Real Time Messaging Protocol", Adobe Systems Incorporated, 2012.

[119] Helix Universal Media Server. Online: [http://www.realnetworks.com/helix/streaming-media-server/].

[120] Wowza Media Server. Online: [http://www.wowza.com/media-server].

[121] P. Romaniak, "Towards Realization of a Framework for Integrated Video Quality of Experience Assessment", IEEE INFOCOM Workshops 2009, pp. 1-2, Rio de Janeiro, 2009.

[122] S-F. Huang, "Video Classification and Adaptive QoP/QoS Control for Multiresolution Video Applications on IPTV", International Journal of Digital Multimedia Broadcasting, 2012.

[123] M. Fiedler, H-J. Zepernick, L. Lundberg, P. Arlos, M.I. Petterson, "QoE-based Cross-Layer Design of Mobile Video Systems: Challenges and Concepts", In Proceedings of the 2009 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF '2009), pp. 1-4, 2009.

[124] K. Lakshminarayanan, S. Seshan, P. Steenkiste, "Undestanding 802.11 performance in heterogeneous environments", Proceeding HomeNets '11: Proceedings of the 2nd ACM SIGCOMM workshop on Home networks, pp. 43-48, New York, 2011.

[125] D. Halperin, W. Hu, A. Sheth, D. Wetherall, "Predictable 802.11 packet delivery from wireless channel measurements", ACM SIGCOMM Computer Communication Review – SIGCOMM '11, Vol. 41, No. 4, pp. 159-170, New York, 2011.

[126] L. Xu, F. Yang, Y. Jiang, L. Zhang, C. Feng, N. Bao, "Variation of Received Signal Strength in Wireless Sensor Network", 2011 3rd International Conference on Advanced Computer Control (ICACC), pp. 151-154, 2011.

[127] A.T. Parameswaran, M.I. Husain, S. Upadhyaya, "Is RSSI a Reliable Parameter in Sensor Localization Algorithms – An Experimental Study", 2009 IEEE 28th International Symposium On Reliable Distributed Systems (SRDS), New York, 2009.

[128] ITU-T Rec. G.114 (05/03), "One-way transmission time", 2003.

[129] B.O. Szuprowicz, "Multimedia Networking", McGraw-Hill Computing, 1995.

[130] Y. Chen, T. Farley, N. Ye, "QoS Requirements of Network Applications on the Internet", Journal Information Knowledge Systems Management, IOS Press, Vol. 4, No. 1, pp. 55-76, Amsterdam, 2004.

[131] L. Hanzo, P.J. Cherriman, J. Streit, "Video Compression and Communications, From Basics to H.261, H.263, H.264, MPEG4 for DVB and HSDPA-Style Adaptive Turbo-Transceivers, 2nd Edition", Wiley-IEEE Press, Chichester, England, 2007.

[132] J.F. Kurose, K.W. Ross, "Computer Networking: A Top-Down Approach, 6th Edition", Addison-Wesley, 2012.

[133] P. Read, M-P. Meyer, "Restoration of motion picture film", Series in Conservation and Museology, Butterworth-Heinemann, 2000.

[134] A.R. Bentivoglio, S.B. Bressman, E. Cassetta, D. Carretta, P. Tonali, A. Albanese, "Analysis of blink rate patterns in normal subjects", Movement Disorders, Vol. 12, No. 6, pp. 1028-1034, 1997.

[135] H.R. Schiffman, "Sensation and Perception. An Integrated Approach - 5th edition", John Wiley and Sons, New York, 2001.

[136] J. Nielsen, "Usability Engineering", Morgan Kaufmann, 1993.

[137] M. Green, "How Long Does It Take to Stop? Methodological Analysis of Driver Perception-Brake Times", Transportation Human Factors, Vol. 2, No. 3, pp. 195–216, 2000.

[138] L. Grüne, J. Pannek, "Nonlinear Model Predictive Control: Theory and Algorithms (Communications and Control Engineering)", Springer-Verlag, London, 2011.

[139] G.G. Malinietski, "Fundamentos matemáticos de la sinergética: caos, estructuras y simulación por ordenador", Editorial URSS, Moscú, 2005.

[140] M. Fiedler, K. Tutschku, P. Carlsson, A. Nilsson, "Identification of performance degradation in IP networks using throughput statistics", In Proceedings of the 18th International Teletraffic Congress (ITC-18), pp. 399– 407, Berlin, 2003.

[141] K.M. Passino, "Intelligent Control: An Overview of Techniques", In T. Samad (Ed.) Perspectives in Control Engineering: Technologies, Applications, and New Directions, Wiley-IEEE Press, pp. 104-133, New Jersey, 2001.

[142] T.M. Mitchell, "Machine Learning", McGraw-Hill Science/Engineering/Math, 1997.

[143] A. Suárez, M. La-Menza, E. Macías, "Recuperación Automática de Sesiones de Streaming en Teléfonos Móviles", VI Jornadas de Ingeniería Telemática, pp. 43-50, Málaga, 2007.

[144] RealNetworks, "Using RTSP with Firewalls, Proxies, and Other Intermediary Network Devices, version 2.0/rev.2", RealNetworks, Inc., 1998.

[145] A. Suárez, E.M. Macías, J. Martín, Y. Gutiérrez, M. Gil, "Light Protocol and Buffer Management for Automatically Recovering Streaming Sessions in WiFi Mobile Telephones", Mobile Ubiquitous Computing Systems, Services and Technologies (UBICOMM2008), pp. 70-76, Valencia, 2008.

[146] R.K.P. Mok, E.W.W. Chan, R.K.C. Chang, "Measuring the Quality of Experience of HTTP Video Streaming", 2011 IFIP/IEEE International Symposium on Integrated Network Management (IM), pp. 485-492, 2011.

[147] C. Perkins, "RTP: Audio and Video for the Internet", Addison-Wesley Professional, 2003.

[148] N. Vlassis, “A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence”, Morgan and Claypool Publishers, 2007.

[149] Y. Shoham, K. Leyton-Brown, “Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations”, Cambridge University Press, 2009.

[150] H.S. Nwana, “Software Agents: An Overview”, The Knowledge Engineering Review, Vol. 11, No. 3, pp. 205-244, 1996.

[151] C. Hewitt, “Viewing Control Structures as Patterns of Passing Messages”, Artificial Intelligence, Vol. 8, No. 3, pp. 323-364, 1977.

[152] Y. Shoham, “Agent-oriented programming”, Journal of Artificial Intelligence, Vol. 60, No. 1, pp. 51-92, 1993.

[153] N.R. Jennings, M.J. Wooldridge, “Applications of Intelligent Agents”, In N.R. Jennings, M.J. Wooldridge (Ed.) “Agent Technology Foundations, Applications, and Markets”, Springer-Verlag, pp. 3-28, Berlin, 2002.

[154] R.A. Belecheanu, S. Munroe, M. Luck, T. Payne, T. Miller, P. McBurney, M. Pěchouček, “Commercial Applications of Agents: Lessons, Experiences and Challenges”, In Proceedings of the 5th International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS’06), pp. 1549-1555, 2006.

[155] G. Ali, N.A. Shaikh, A.W. Shaikh, “A Research Survey of Software Agents and Implementation Issues in Vulnerability Assessment and Social Profiling Models”, Australian Journal of Basic and Applied Sciences, Vol. 4, No. 3, pp. 442-449, 2010.

[156] H. Hamidi, K. Mohammadi, “Modeling Fault Tolerant and Secure Mobile Agent Execution in Distributed Systems”, In V. Sugumaran (Ed.) Distributed Artificial Inteligence, Agent Technology, and Collaborative Applications, pp. 132-146, Information Science Reference, Hershey, PA, 2009.

[157] S. Franklin, A. Graesser, “Is it an Agent, or Just a Program?: A Taxonomy for Autonomous Agents”, In Proceedings of the Workshop on Intelligent Agents III, Agent Theories, Architectures, and Languages (ECAI '96), pp. 21-35, 1996.

[158] M. Wooldridge, N.R. Jennings, “Intelligent agents: theory and practice”, The Knowledge Engineering Review, Vol. 10, No. 2, pp. 115-152, 1995.

[159] J. Bates, “The role of emotion in believable agents”, Magazine Communications of the ACM, Vol. 37, No. 7, pp. 122-125, 1994.

[160] M.C. Huebscher, J.A. McCann, “A Surrey of Autonomic Computing – Degrees, Models, and Applications”, ACM Computing Surveys, Vol. 40, No. 3, Article 7, 2008.

[161] S. Russell, P. Norvig, “Artificial Intelligence: A Modern Approach (3rd Edition) ”, Prentice Hall, New Jersey, 2010.

[162] K.T. Strongman, “The Psychology of Emotion: From Everyday Life to Theory, Fifth edition”, John Wiley & Sons Ltd, Chichester, England, 2003.

[163] C.E. Georgakarakou, A.A. Economides, "Software Agent Technology: An Overview", In P.F. Tiako (Ed.) Software Applications: Concepts, Methodologies, Tools, and Applications, pp. 128-151, IGI-Global, USA, 2009.

[164] J.M. Bradshaw, "An Introduction to Software Agents", In J.M. Bradshaw (Ed.) Software Agents, AAAI Press/The MIT Press, pp. 3-46, California, 1997.

[165] C.M. Maca, M.J. North, "Tutorial on agent-based modelling and simulation", Journal of Simulation Vol. 4, No. 3, pp. 151-162, 2010.

[166] S.E. Page, J.H. Miller, "Complex Adaptive Systems: An Introduction to Computational Models of Social Life", Princeton University Press, 2007.

[167] M.J. North, N.T. Collier, J. Ozik, E.R. Tatara, C.M. Macal, M. Bragen, P. Sydelko, "Complex adaptive systems modeling with Repast Simphony", Complex Adaptive Systems Modeling, Vol. 1, n. 3, 2013.

[168] F. Neri, "Software agents as versatile simulation tool to model complex systems", Journal World Scientific and Engineering Academy and Society (WSEAS) Transactions on Information Science and Applications, Vol. 7, No. 5, pp. 609-618, 2010.

[169] G. Rimmel, M. Clement, M. Runte, "Intelligent Software Agents: Implicatons for Marketing in eCommerce", In Proceedings of the 22nd Information Systems Research Seminar in Scandinavia (IRIS 22): "Enterprise Architectures for Virtual Organisations", Vol. 1, pp. 203-218, Finland, 1999.

[170] C.A. Knoblock, "Building software agents for planning, monitoring, and optimizing travel", In A.J. Frew (Ed.) Information and Communications Technologies in tourism 2004, Proceedings of the 11th International Conference in Cairo, Egypt, 2004, pp. 1-15, Viena, 2004.

[171] C.A. Kumar, S. Srinivas, "On Adopting Software Agents for Distributed Digital Libraries", DESIDOC Bulletin of Information Technology, Vol. 24, No. 3, pp. 3-8, 2004.

[172] I. Satoh, "Software agents for ambient intelligence", 2004 IEEE International Conference on Systems, Man & Cybernetics, Vol. 2, pp. 1147-1152, 2004.

[173] T. Dimitrova, "Software Agents for Project Management in a Database Environment", In Proceedings of the 2nd International Scientific Conference "Computer Science'2005", Part 2, pp. 172-177, Chalkidiki, Greece, 2005.

[174] I.T. Hawryszkiewycz, "Software Agents for Managing Learning Plans", Journal of Issues in Informing Science and Information Technology, Vol. 3, pp. 269-277, 2006.

[175] M. Lluch-Ariet, F. Estanyol, M. Mier, C. Delgado, H. González-Vélez, T. Dalmas, M. Robles, C. Sáez, J. Vicente, S. Van Huffel, J. Luts, C. Arús, A.P. Candiota Silveira, M. Julià-Sapé, A. Peet, A. Gibb, Y. Sun, B. Celda, M.C. Martínez Bisbal, G. Valsecchi, D. Dupplaw, B. Hu, P. Lewis, "HealthAgents: Agent-Based Distributed Decision Support System for Brain Tumour Diagnosis and Prognosis", In R.

Annicchiarico, U. Cortés, C. Urdiales (Eds.) Agent Technology and e-Health, pp. 5-24, Birkhäuser Verlag, Basel, Switzerland, 2007.

[176] M. Nguyen-Duc, Z. Guessoum, O. Marin, J-F. Perrot, J-P. Briot, V. Duong, "Towards a Reliable Air Traffic Control (Short Paper)", In Proceedings of the 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '08), pp. 101-104, Estoril, Portugal, 2008.

[177] K. Smarsly, D. Hartmann, "Real-Time Monitoring of Wind Energy Converters Based on Software Agents", In K. Gürlebeck, C. Könke (Eds.) Proceedings of the 18th International Conference on the Application of Computer Science and Mathematics in Architecture and Civil Engineering (IKM 2009), Weimar, Germany, 2009.

[178] I.N. Athanasiadis, P.A. Mitkas, "A Methodology for Developing Environmental Information Systems with Software Agents", In U. Cortés, M. Poch (Eds.) Advanced Agent-Based Environmental Management Systems, pp. 119-137, Birkhäuser Verlag, Basel, Switzerland, 2009.

[179] J. Cartlidge, C. Szostek, M. De Luca, D. Cliff, "Too fast too furious: faster financial-market trading agents can give less efficient markets", Proceedings of the 4th International Conference on Agents and Artificial Intelligence (ICAART 2012), Vol. 2, pp. 126-135, Portugal, 2012.

[180] A.S.M. De Franceschi, J.M. Barreto, M. Roisenberg, "Autonomous Software Agents for Computer Network Management", In IEEE International Conference on Telecommunications 2000 (ICT2000), Acapulco, Mexico, 2000.

[181] D. Dasgupta, H. Brian, "Mobile security agents for network traffic analysis", In Proceedings of the DARPA Information Survivability Conference & Exposition II, 2001 (DISCEX '01), Vol. 2, pp. 332-340, Anaheim, CA, 2001.

[182] S. Makki, S.V. Wunnava, "Application of Mobile Agents in Managing the Traffic in the Network and Improving the Reliability and Quality of Service", IANG International Journal of Computer Science, Vol. 32, No. 4, 2006.

[183] H. Khazaei, J. Mišić, V.B. Mišic, "Mobile Software Agents for Wireless Network Mapping and Dynamic Routing", 2010 IEEE 30th Intenational Conference on Distributed Computing Systems Workshops (ICDCSW), pp. 240-246, Genova, 2010.

[184] T. Yoshimura, T. Ohya, T. Kawahara, M. Etoh, "Rate and robustness control with RTP monitoring agent for Mobile multimedia streaming", 2002 IEEE International Conference on Communications (ICC 2002), Vol. 4, pp. 2513-2517, 2002.

[185] A. Vrančić, K. Jurasović, M. Kušek, G. Ježić, K. Tržec, "Service Provisioning in Telecommunication Networks using Software Agents and Rule-based Approach", In S. Golubić, B. Mikac, V. Hudek (Eds.) Proceedings of the 30th International Convention MIPRO 2007, pp. 159-164, Opatija, Croatia, 2007.

[186] F. Bellifemine, G. Caire, A. Poggi, G. Rimassa, "JADE, A White Paper", Journal of Telecom Italia Lab, Vol. 3, No. 3, pp. 6-19, 2003.

[187] S.O. Hansson, "Decision Theory: A Brief Introduction", Department of Philosophy and the History of Technology of the Royal Institute of Technology (KTH), Stockholm, 1994.

[188] F. Aguiar, "Teoría de la decisión e incertidumbre: modelos normativos y descriptivos", Empiria: Revista de Metodología de Ciencias Sociales, No. 8, pp. 139-160, 2004.

[189] S. Ríos García, S. Ríos Insúa, "La teoría de la decisión de Pascal a Von Neumann", Historia de la Matemática, pp. 11-42, 1998.

[190] W.C. Stirling, "Satisficing Games and Decision Making: with applications to engineering and computer science", Cambridge University Press, Cambridge, United Kingdom, 2003.

[191] S. Das, "Foundations of Decision-Making Agents: Logic, Porbability and Modality", World Scientific Publishing/Imperial College Press, Singapore, 2008.

[192] D.M. Bourg, G. Seeman, "AI for Game Developers", O'Reilly, Sebastopol, CA, 2004.

[193] D. Charles, C. Fyfe, D. Livingstone, S. McGlinchey, "Biologically Inspired Artificial Intelligence for Computer Games", IGI Publishing, Hershey, PA, 2008.

[194] D.C. Skinner, "Introduction to Decision Analysis, 3rd Edition", Probabilistic Publishing, Sugar Land, Texas, 2009.

[195] H.A. Simon, "The New Science of Management Decision", Harper and Row, New York, NY, 1960.

[196] A. Ligeza, "Logical Foundations for Rule-Based Systems (Studies in Computational Intelligence)", Springer, Berlin, 2006.

[197] E.L. Post, "Formal Reductions of the General Combinatorial Decision Problem", American Journal of Mathematics, Vol. 65, No. 2, pp. 197-215, 1943.

[198] P. Dayan, L.F. Abbott, "Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems", The MIT Press, Massachusetts, 2005.

[199] S. Haykin, "Neural Networks and Learning Machines (3rd Edition)", Prentice Hall, New Jersey, 2008.

[200] J. Mańdziuk, R. Mikołajczak, "Chaotic time series prediction with feed-forward and recurrent neural nets", Control and Cybernetics, Vol. 31, No. 2, pp. 383-406, 2002.

[201] C. Jiang, F. Song, "Sunspot Forecasting by Using Chaotic Time-series Analysis and NARX Network", Journal of Computers, Vol. 6, No. 7, pp. 1424-1429, 2011.

[202] R.J. Frank, N. Davey, S.P. Hunt, "Time Series Prediction and Neural Networks", Journal of Intelligent and Robotic Systems, Vol. 31, No. 1-3, pp. 91-103, 2001.

[203] S. Tronci, M. Giona, R. Baratti, "Reconstruction of chaotic time series by neural models: a case study", Neurocomputing, Vol. 55, No. 3-4, pp. 581-591, 2003.

[204] T. Most, "Approximation of complex nonlinear functions by means of neural networks", In Proceedings of the 2nd Weimar Optimization and Stochastic Days (WOST 2005), Weimar, 2005.

[205] J.H. Holland, "Adaptation in Natural and Artificial Systems", University of Michigan Press, Ann Arbor, Michigan, 1975.

[206] S.D. Sloan, R.W. Saw, J.L. Sluss, M.P. Tull, J.P. Havlicek, "Genetic Algorithm Forecasting for Telecommunications Products", In C.H. Dagli (Eds.) Intelligent Engineering Systems Through Artificial Neural Networks (Volume 9), pp. 361-368, ASME Press, New York, USA, 1999.

[207] C.M. Kishtawal, S. Basu, F. Patadia, P.K. Thapliyal, "Forecasting summer rainfall over India using genetic algorithm", Geophysical Research Letters, Vol. 30, No. 23, 2003.

[208] V. Babovic, S.A. Sannasiraj, E.S. Chan, "Error correction of a predictive ocean wave model using local model approximation", Journal of Marine Systems, Vol. 53, No. 1-4, pp. 1-17, 2005.

[209] P. Flores, L. Burtseva, L.B. Morales, "Self Adaptive Genetic Algorithms for Automated Linear Modelling of Time Series", In S. Gao (Ed.) Bio-Inspired Computational Algorithms and Their Applications, InTech, pp. 213-234, 2012.

[210] L.A. Zadeh, "Fuzzy sets", Information and Control, Vol. 8, No. 3, pp. 338-353, 1965.

[211] E.H. Mamdani, S. Assilian, "An experiment in linguistic synthesis with a fuzzy logic controller", International Journal of Man-Machine Studies, Vol. 7, No. 1, pp. 1–13, 1975.

[212] T. Takagi, M. Sugeno, "Fuzzy identification of systems and its applications to modeling and control", IEEE transactions on Systems, Man, and Cybernetics, Vol. 15, No. 1, pp. 116–132, 1985.

[213] L.-X. Wang, J.M. Mendel, "Generating Fuzzy Rules by Learning from Examples", IEEE Transactions on Systems, Man, and Cybernetics, Vol. 22, No. 6, 1992.

[214] Y. Wang, Y. Chen, "A comparison of Mamdani and Sugeno fuzzy inference systems for chaotic time series prediction", In 2012 2nd International Conference on Computer Science and Network Technology (ICCSNT), pp. 438-442, China, 2012.

[215] N. Boumella, S. Iqbal, M. Boulemden, K. Djouani, "Designing a Global Fuzzy Logic Forecaster for Chaotic Time Series", Annals of DAAAM for 2011 & Proceedings of the 22nd International DAAAM Symposium, Volume 22, No. 1, pp. 1473-1474, 2011.

[216] E. Chin, D. Chieng, V. Teh, M. Natkaniec, K. Loziak, J. Gozdecki, "Wireless link prediction and triggering using modified Ornstein–Uhlenbeck jump diffusion process", Wireless Networks, Vol. 20, No. 3, pp. 379-396, 2014.

[217] A. Basharat, M. Shah, "Time series prediction by chaotic modeling of nonlinear dynamical systems", In 2009 IEEE 12th International Conference on Computer Vision, pp. 1941-1948, Japan, 2009.

[218] C. Casado, R. Colomo, "Un breve recorrido por la concepción de las emociones en la Filosofía Occidental", A Parte Rei. Revista de Filosofía, Vol. 46, Madrid, 2006.

[219] R. Descartes, "Discurso del Método", trad. R. Frondizi, Alianza Editorial, Madrid, 1999.

[220] F. Nietzsche, "Más allá del bien y del mal", trad. A. Sánchez Pascual, Alianza Editorial, Madrid, 1997.

[221] B. Pascal, "Pensamientos", trad. X. Zubiri Apalategui, Alianza Editorial, Madrid 2004.

[222] F. Nietzsche, "Crepúsculo de los ídolos", trad. A. Sánchez Pascual, Alianza Editorial, Madrid, 1998.

[223] B. Pinna, "New Gestalt Principles of Perceptual Organization: An Extension of Grouping to Shape and Meaning", Gestalt Theory – An International Multidisciplinary Journal, Vol. 32, No. 1, pp. 11-78, 2010.

[224] R. de Sousa, "The Rationality of Emotion", MIT Press, Massachusetts, 1990.

[225] M. Akerman, "Gestalt Principles. 12 Examples. A composition".

[226] V.M. Simón, "La participación emocional en la toma de decisiones", Psicothema, Vol. 9, No. 2, pp. 365-376, 1997.

[227] J.M. Martínez-Selva, J.P. Sánchez-Navarro, A. Bechara, F. Román, "Mecanismos cerebrales de la toma de decisiones", Rev Neurol, Vol. 42, No. 7, pp. 411-418, 2006.

[228] D. Tranel, A.R. Damasio, "Neuropsychology and behavioral neurology", In J.T. Cacioppo, L.G. Tassinary, G.G. Berntson (Eds.) Handbook of psychophysiology, 2 ed., Cambridge University Press, pp. 119-141, New York, 2000.

[229] D. Keltner, J.J. Gross, "Functional Accounts of Emotions", Psychology Press: Cognition and Emotion, Vol. 13, No. 5, pp. 467-480, 1999.

[230] A. Bechara, "Risky business: emotion, decision-making, and addiction", J Gambl Stud, Vol. 19, pp. 23-51, 2003.

[231] A. Verdejo, F. Aguilar de Arcos, M. Pérez-García, "Alteraciones de los procesos de toma de decisiones vinculados al cortex prefrontal ventromedial en pacientes drogodependientes", Revista de Neurología, Vol. 38, pp. 601-606, 2004.

[232] R.M.A. Nelissen, A.J.M. Dijker, N.K. De Vries, "Emotions and goals: Assessing relations between values and emotions", Psychology Press: Cognition and Emotion, Vol. 21, No. 4, pp. 902-911, 2007.

[233] U. Hess, "The experience of emotion: Situacional influences on the elicitation and experience of emotions", In A. Kaszniak (Ed.) Emotions, Qualia, and Consciousness, World Scientific Publishing, pp. 386-396, Singapore, 2001.

[234] A.R. Damasio, "El error de Descartes", Crítica, Barcelona, 2006.

[235] A. Bechara, H. Damasio, D. Tranel, A.R. Damasio, "The Iowa Gambling Task and the somatic marker hypothesis: some questions and answers", Trends Cohn Sci, Vol. 9, pp. 159-162, 2005.

[236] A. Bechara, H. Damasio, "Deciding advantageously before knowing the advantageous strategy", Science, Vol. 275, pp. 1293-1295, 1997.

[237] A. Bechara, "Decision making, impulse control and loss of willpower to resist drugs: a neurocognitive perspective", Nature Neuroscience, Vol. 8, pp. 1458-1463, 2005.

[238] J. LeDoux, "The Emotional Brain: The Mysterious Underpinnings of Emotional Life", Simon & Schuster, New York, 1998.

[239] J.D. Velásquez, "A Computational Framework for Emotion-Based Control", In: Proceeding of the Grounding Emotions in Adaptive Systems Workshop, SAB '98, Zurich, 1998.

[240] R.R. Murphy, C.L. Lisetti, R. Tardif, L. Irish, A. Gage, "Emotion-Based Control of Cooperating Heterogeneous Mobile Robots", IEEE Transactions on Robotics and Automation, Vol. 18, No. 5, pp. 744-757, 2002.

[241] M. Bedia, "Arquitecturas emocionales en inteligencia artificial: una propuesta unificadora", Revista Electrónica Teoría de la Educación: Educación y Cultura en la Sociedad de la Información, Vol. 7, No. 2, pp. 156-167, 2006.

[242] S. Tomkins, "Affect Theory", In K. Scherer, P. Ekman (Eds.), Approaches to Emotion, Lawrence Erlbaum Associates, Hillsdale, NJ, 1984.

[243] H. A. Simon. "Affect and Cognition: Comments", In M.S. Clark, S.T. Fiske (Eds.) The Seventeenth Annual Carnegie Symposium on Cognition: Affect and Cognition, Lawrence Erlbaum Associates, pp. 333-342, London, 1982.

[244] A. Sloman, "Motives mechanisms and emotions", Cognition and Emotion, Vol. 1, No. 3, pp. 217-234, 1987.

[245] N. Frijda, "The emotions. Sudies in Emotion and Social Interactions", Cambridge University Press, Cambridge, UK, 1986.

[246] R.S. Lazarus, "On the primacy of cognition", American Psychologist, Vol. 39, No. 2, pp. 124-129, 1984.

[247] P. Petta, "The role of emotions in a tractable architecture for situated cognizers", In R. Trappl, P. Petta, S. Payr (Eds.), Emotions in Humans and Artifacts, MIT Press, pp. 251-288, Massachusetts, 2003.

[248] B.R. Steunebrink, M. Dastani, J-J.Ch. Meyer, "A Logic of Emotions for Intelligent Agents", Proceedings of the Twenty-Second Conference on Artificial Intelligence (AAAI'07), pp. 142-147, Vancouver, Canada, 2007.

[249] M.F. Bear, B.W. Connors, M.A. Paradiso, "Neuroscience: Exploring the Brain, 3rd Edition", Lippincott Williams and Wilkins, 2007.

[250] J. Horgan, "The Forgotten Era of Brain Chips", Scientific American, Vol. 293, No. 4, pp. 66-73, 2005.

[251] S.A. Huettel, A.W. Song, G. McCarthy, "Functional Magnetic Resonance Imaging (2nd Edition)", Sinauer Associates, 2009.

[252] M. Libenson, "Practical Approach to Electroencephalography", Saunders Elsevier, 2009.

[253] P. Hansen, M. Kringelbach, R. Salmelin, "MEG: An Introduction to Methods", Oxford University Press, New York, 2010.

[254] R.B. Silberstein, M.A. Schier, A. Pipingas, J. Ciorciari, S.R. Wood, D.G. Simpson, "Steady-State Visually Evoked Potential topography associated with a visual vigilance task", Brain Topography, Vol. 3, No. 2, pp. 337-347, 1990.

[255] A. Bojko, "Eye Tracking the User Experience: A Practical Guide to Research", Rosenfeld Media, 2013.

[256] W. Boucsein, "Electrodermal Activity, 2nd Edition", Springer, 2012.

[257] T. Trappenberg, "Fundamentals of Computational Neuroscience, 2nd Edition", Oxford University Press, New York, USA, 2010.

[258] E.M. Izhikevich, "Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting", The MIT Press, Massachusetts, 2007.

[259] F. Gabbiani, S.J. Cox, "Mathematics for Neuroscientists", Academic Press, 2010.

[260] H. Markram, "The Blue Brain Project", Nature Reviews Neuroscience, Vol. 7, pp. 153-160, 2006.

[261] H. Markram, K. Meier, T. Lippert, S. Grillner, R. Frackowiak, S. Dehaene, A. Knoll, H. Sompolinsky, K. Verstreken, J. DeFelipe, S. Grant, J.-P. Changeux, A. Saria, "Introducing the Human Brain Project", Proceedings of the 2nd European Future Technologies Conference and Exhibition 2011 (FET 11), Vol. 7, pp. 39-42, 2011.

[262] M.S. Gazzaniga (Ed.), "The Cognitive Neurosciences, 4th Edition", A Bradford Book, Massachussets, 2009.

[263] P.W. Glimcher, E. Fehr (Eds.), "Neuroeconomics: Decision Making and the Brain, 2nd Edition", Academic Press, 2014.

[264] L. Zurawicki, "Neuromarketing: Exploring the Brain of the Consumer", Springer-Verlag Berlin Heidelberg, 2010.

[265] P.E. Bermejo, R. Dorado, M.A. Zea-Sevilla, V. Sánchez, "Neuroanatomía de las decisiones financieras", Neurología, Vol. 26, No. 3, pp. 173-181, 2011.

[266] S.L. Bressler, V. Menon, "Large-scale brain networks in cognition: emerging methods and principles", Trends in Cognitive Sciences, Vol. 14, No. 6, pp. 277-290, 2010.

[267] D. Barber, "Bayesian Reasoning and Machine Learning", Cambridge University Press, New York, 2012.

[268] J. Pearl, "Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference", Morgan Kaufmann, Massachussets, 1988.

[269] D. George, J. Hawkins, "A Hierarchical Bayesian Model of Invariant Pattern Recognition in the Visual Cortex", Proceedings of the 2005 IEEE International Joint Conference on Neural Networks (IJCNN '05), Vol. 3, pp. 1812-1817, Montreal, Canada, 2005.

[270] E.T. Carlson, R.J. Rasquinha, K. Zhang, C.E. Connor, "A Sparse Object Coding Scheme in Area V4", Current Biology, Vol. 21, No. 4, pp. 288-293, 2011.

[271] J. Hawkins, S. Blakeslee, "On Intelligence", Times Books, New York, 2004.

[272] M. Minsky, "The Society of Mind", Simon and Schuster, New York, 1988.

[273] H. Haken, "Brain Dynamics: An Introduction to Models and Simulations, 2nd Edition", Springer, 2008.

[274] D. Kahneman, "Thinking, Fast and Slow", Farrar, Straus and Giroux, New York, 2011.

[275] V.J. Wukmir, "Emoción y Sufrimiento", Labor, Barcelona, 1967.

[276] K. Oatley, "Best Laid Schemes: The Psychology of the Emotions", Cambridge University Press, New York, 1992.

[277] M.J. Reeve, "Motivación y Emoción", McGraw-Hill, Madrid, 1994.

[278] P. Ekman, "Emotions Revealed, 2nd Edition: Recognizing Faces and Feelings to Improve Communication and Emotional Life", Henry Holt and Company, New York, 2007.

[279] K.R. Scherer, "Neuroscience Projections to Current Debates in Emotion Psychology", Cognition and Emotion, Vol. 7, No. 1, pp. 1-41, 1993.

[280] W. Wundt, "Grundriss der Psychologie", Leipzig: Wilhelm Engelmann, Germany, 1896.

[281] R.B. Zajonc, “Feeling and thinking: Preferences need no inferences”, American Psychologist, Vol. 35, No. 2, pp. 151-175, 1980.

[282] S-H. Cha, “Taxonomy of Nominal Type Histogram Distance Measures”, Proceedings of the American Conference on Applied Mathematics (MATH '08), pp. 325-330, Cambridge, Massachusetts, USA, 2008.

[283] P. Ekman, “Universals and Cultural Differences in Facial Expressions of Emotion”, In J. Cole (Ed.) Nebraska Symposium on Motivation, 1971, University of Nebraska Press, Vol. 19, pp. 207-282, Lincoln, Nebraska, 1972.

[284] P. Ekman, “Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage, 3rd Edition”, W.W.Norton and Company, New York, 2009.

[285] V. Ramos Linares, J.A. Piqueras Rodríguez, A.E. Martínez González, L.A. Oblitas Guadalupe, “Emoción y Cognición: Implicaciones para el Tratamiento”, Terapia Psicológica, Vol. 27, No. 2, pp 227-237, 2009.

[286] F. Palmero, “Emoción. Breve reseña del papel de la cognición y el estado afectivo”, Revista Electrónica de Motivación y Emoción, Vol. 2, No. 2-3, 1999.

[287] F. Palmero, C. Gómez, C. Guerrero, A. Carpi, “Manual de prácticas de motivación y emoción”, Publicacions de la Universitat Jaume I, Castelló de la Plana, 2010.

[288] D.G. Myers, “Exploring Psychology, 9th Edition”, Worth Publishers, New York, 2012.

[289] D. Broadbent, “Perception and Communication”, Pergamon Press, London, 1958.

[290] R.C. Atkinson, R.M. Shiffrin, “Human memory: A proposed system and its control processes”, In K.W. Spence, J.T. Spence (Eds.) The Psychology of Learning and Motivation: Advances in Research and Theory, Academic Press, Vol. 2, pp. 89-195, New York, 1968.

[291] A.D. Baddeley, G. Hitch, “Working memory”, In G.H. Bower (Ed.) The Psychology of Learning and Motivation: Advances in Research and Theory, Academic Press, Vol. 8, pp. 47-89, New York, 1974.

[292] M.B. Howes, G. O'Shea, “Human Memory: A Constructivist View”, Academic Press, 2014.

[293] D. Páez Rovira, A.J. Carbonero Martínez, “Afectividad, cognición y conducta social”, Psicothema, vol. 5, Suplemento, pp. 133-150, 1993.

[294] J.R. Reynolds, J.M. Zacks, T.S. Braver, “A Computational Model of Event Segmentation From Perceptual Prediction”, Cognitive Science, Vol. 31, pp. 613-643, 2007.

[295] L. Festinger, “A theory of cognitive dissonance”, Stanford University Press, Stanford, California, 1957.

[296] H. Ebbinghaus, “Memory: A Contribution to Experimental Psychology”, Teachers College, Columbia University, New York, 1913.

[297] S. Pinker, “The Blank Slate: The Modern Denial of Human Nature”, Penguin Books, 2003.

[298] E.B. Goldstein, “Sensation and Perception, 9th Edition”, Wadsworth-Cengage Learning, Belmont, California, 2013.

[299] G.T. Fechner, “Elemente der Psychophysik”, Breitkopf und Härtel, Leipzig, 1860.

[300] S.S. Stevens, “On the psychophysical law”, Psychological Review, Vol. 64, No. 3, pp. 153-181, 1957.

[301] S.S. Stevens, “To Honor Fechner and Repeal His Law”, Science, Vol. 133, No. 3446, pp. 80-86, 1961.

[302] C. Gros, “Complex and Adaptive Dynamical Systems: A Primer”, Springer, 2008.

[303] MacBook Air (13-inch, Early 2014) – Technical Specifications. Online: [http://support.apple.com/kb/SP700].

[304] Intel Core i5-4260U Processor. Online: [http://ark.intel.com/products/75030].

[305] OS X Yosemite - Technical Specifications. Online: [http://support.apple.com/kb/SP711].

[306] Java SE Downloads. Online: [http://www.oracle.com/technetwork/java/javase/downloads/index.html].

[307] Cisco WAP200 Wireless-G Access Point: PoE/RangeBooster Cisco Small Business Access Points. Online: [http://www.cisco.com/c/en/us/products/collateral/wireless/wap200-wireless-g-access-point-poe-rangebooster/data_sheet_c78-501966.html].

[308] Darwin Streaming Server & Proxy. Online: [http://dss.macosforge.org/].

[309] QuickTime Broadcaster. Online: [http://support.apple.com/kb/DL764].

[310] Telestream Wirecast. Online: [http://www.telestream.net/wirecast/overview.htm].

[311] F. Bellifemine, G. Caire, D. Greenwood, “Developing Multi-Agent Systems with JADE”, Wiley, 2007.

[312] “FIPA Abstract Architecture Specification”, Foundation for Intelligent Physical Agents, 2002.

[313] Giovanni Caire, Federico Pieri, “LEAP User Guide”, Telecom Italia Lab, 2011.

[314] JADE – Java Agent DEvelopment Framework. Online: [http://jade.tilab.com/].

[315] Zeus Agent Toolkit. Online: [http://sourceforge.net/projects/zeusagent/].

[316] Aglets. Online: [http://aglets.sourceforge.net/].

[317] SeMoA. Online: [http://semoa.sourceforge.net/].

[318] Microsoft .NET. Online: [http://www.microsoft.com/net].

[319] M. Nikraz, G. Caire, P.A. Bahri, "A Methodology for the Analysis and Design of Multi-Agent Systems using JADE", International Journal of Computer Systems Science & Engineering, Vol. 21, No. 2, pp. 99-116, 2006.

[320] Unified Modeling Language (UML). Online: [http://www.uml.org/].

[321] The FIPA Agent UML (AUML). Online: [http://www.auml.org/].

[322] RTP Java Library. Online: [http://sourceforge.net/projects/jlibrtp/].

[323] jSDP. Online: [http://sourceforge.net/projects/jsdp/].

[324] JFreeChart. Online: [http://www.jfree.org/jfreechart/].

[325] Wireshark. Online: [http://www.wireshark.org/].

[326] Eclipse. Online: [http://www.eclipse.org/].

[327] Protégé. Online: [http://protege.stanford.edu/].

[328] OntologyBeanGenerator. Online: [http://protegewiki.stanford.edu/wiki/OntologyBeanGenerator].

[329] R: The R Project for Statistical Computing. Online: [http://www.r-project.org/].

[330] RStudio. Online: [http://www.rstudio.com/].

[331] "FIPA ACL Message Structure Specification", Foundation for Intelligent Physical Agents, 2002.

[332] T. Finin, J. Weber, G. Wiederhold, M. Genesereth, R. Fritzson, D. McKay, J. McGuire, R. Pelavin, S. Shapiro, C. Beck, "Specification of the KQML Agent-Communication Language – plus example agent policies and architectures", The DARPA Knowledge Sharing Initiative External Inerfaces Working Group, 1993.

[333] K. Spalek, M. Fastenrath, S. Ackermann, B. Auschra, D. Coynel, J. Frey, L. Gschwind, F. Hartmann, N. van der Maarel, A. Papassotiropoulos, D. de Quervain, A. Milnik, "Sex-Dependent Dissociation between Emotional Appraisal and Memory: A Large-Scale Behavioral and fMRI Study", The Journal of Neuroscience, Vol. 35, No. 3, pp. 920-935, 2015.

[334] T.W. Schmitz, E. De Rosa, A.K. Anderson, "Opposing Influences of Affective State Valence on Visual Cortical Encoding", The Journal of Neuroscience, Vol. 29, No. 22, pp. 7199-7207, 2009.

[335] B. Peters, J. Kaiser, B. Rahm, C. Bledowski, "Activity in Human Visual and Parietal Cortex Reveals Object-Based Attention in Working Memory", The Journal of Neuroscience, Vol. 35, No. 8, pp. 3360-3369, 2015.

[336] S. Shokur, J.E. O'Doherty, J.A. Winans, H. Bleuler, M.A. Lebedev, M.A.L. Nicolelis, "Expanding the primate body schema in sensorimotor cortex by virtual touches of an avatar", PNAS, Vol. 110, No. 37, pp. 15121-15126, 2013.

[337] M. Wimber, A. Alink, I. Charest, N. Kriegeskorte, M.C. Anderson, "Retrieval induces adaptive forgetting of competing memories via cortical pattern suppression", Nature Neuroscience, Vol. 18, No. 4, pp. 582–589, 2015.

[338] A.Y. Sklar, N. Levy, A. Goldstein, R. Mandel, A. Maril, R.R. Hassin, "Reading and doing arithmetic nonconsciously", PNAS, Vol. 109, No. 48, pp. 19614-19619, 2012.

[339] G. de Lavilléon, M.M. Lacroix, L. Rondi-Reig, K. Benchenane, "Explicit memory creation during sleep demonstrates a causal role of place cells in navigation", Nature Neuroscience Vol. 18, No. 4, pp. 493-495, 2015.

[340] Wowza Streaming Engine: RTSP Streaming. Direct RTSP URL: [rtsp://wowzaec2demo.streamlock.net/vod/mp4:BigBuckBunny_115k.mov].

[341] Amazon Elastic Compute Cloud (EC2) – Alojamiento escalable en la nube. Online: [http://aws.amazon.com/es/ec2/].

[342] OpenCL - The open standard for parallel programming of heterogeneous systems. Online: [http://www.khronos.org/opencl/].

[343] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, D. Hassabis, "Human-level control through deep reinforcement learning", Nature, Vol. 518, No. 7540, pp. 529–533, 2015.

[344] "Watson - A System Designed for Answers. The future of workload optimized systems design", IBM White Paper, 2011.

[345] E. Morsella, C.A. Godwin, T.K. Jantz, S.C. Krieger, A. Gazzaley, "Homing in on Consciousness in the Nervous System: An Action-Based Synthesis", Behavioral and Brain Sciences, pp. 1-106, 2015.