CAPÍTULO 7

TAP COMO SISTEMA DISTRIBUIDO

7.1. INTRODUCCIÓN

Las posibilidades de la tecnología ATM con anchos de banda escalables entre los 155 Mbps y varios Gbps le aportan grandes posibilidades para su diseño como sistema distribuido. Sin embargo, algunas de las cuestiones que ya hemos comentado, como la orientación a la conexión de ATM, el control de flujo y de errores fuera de la red, y las retransmisiones extremo-extremo le dan una apariencia muy alejada de la visión clásica de los sistemas distribuidos. Sabemos que los conmutadores presentes en la red pueden descartar células cuando aparecen las congestiones. Sin embargo, cuando estas células forman parte de PDU, la pérdida de una célula es detectada en el destinatario, lo que implica que deberá realizarse la solicitud de retransmisión desde el origen, previa solicitud por parte del destinatario. Este escenario nos sitúa por tanto en una posición en la cual se justifica la necesidad de mecanismos distribuidos en la red que permitan poder solventar los problemas de forma local a donde aparecen. De este modo, la responsabilidad de resolver problemáticas como la congestión, puede ser distribuida a lo largo de toda la red sin necesidad de implicar a los extremos de la conexión, ni tampoco a aquellos tramos de la red que no experimentan congestiones. Precisamente, lo que nos proponemos con TAP es la posibilidad de distribuir los conmutadores que soportan el protocolo de GoS a lo largo de una red VPN, lo que aporta a la tecnología ATM esa característica de sistema distribuido que, en no pocas ocasiones, se le ha reclamado.

Podemos adecuar la definición realizada en [1] sobre lo que es un SD (Sistema Distribuido) y ajustarla a nuestras necesidades concretas, de modo que podemos decir que un SD consiste en un conjunto de dispositivos de cómputo (ordenadores, conmutadores y elementos de interconexión) unidos todos por una red y equipados con un sistema software no centralizado. De este modo, el software no centralizado conduce a la idea de un sistema software distribuido que permite a los elementos hardware del SD coordinar sus actividades y compartir los recursos del sistema (hardware, software y datos). El objetivo final del SD es conseguir que sus usuarios lo vean como un único e integrado entorno de computación, aunque en realidad se trate de diferentes elementos dispersos geográficamente e interconectados por una red.

Los agentes software distribuidos (multiagentes) son una atractiva y emergente tecnología que permite el procesamiento de información distribuida. Los objetivos de estos agentes son los dos principios de la construcción de sistemas complejos: la construcción de agentes múltiples y de mecanismos para la coordinación de agentes de comportamiento independientes. Los agentes son útiles en aplicaciones donde el entorno cambia dinámicamente y su comportamiento inteligente es deseable y hasta esencial (como en la toma de decisiones, predicción y planificación). En estas aplicaciones , estas tareas son realizadas por un conjunto de programas en paralelo pero independientes. Incluso, los entornos en los cuales, por su naturaleza no requieren de sistemas distribuidos, pueden beneficiarse del uso de sistemas multiagente. Estos beneficios son la escalabilidad, robustez y programación más sencilla. En el caso de las comunicaciones, una de estas aplicaciones es la gestión de la red, pero otras pueden ser el routing, monitorización del rendimiento, gestión de conexiones, control de congestiones, etc.

Tradicionalmente, la gestión de los sistemas de telecomunicaciones se ha realizado a través de sistemas centralizados, muy seguros, pero con importantes carencias como la dependencia de funcionamiento de todo

el sistema de elementos concretos de la red, como la escasa escalabilidad y la elevada complejidad, por citar algunas de ellas. La aparición de tecnologías como ATM, que permite la reconfiguración dinámica de los recursos de la red en tiempo de ejecución, ha dado lugar a nuevos servicios que ya han sido comentados en capítulos previos. No obstante, aunque la tecnología ATM ha sido concebida para aportar estos y otros requerimientos, aparece un importante problema como es el resolver la complejidad resultante de estas demandas. Una forma de resolver este problema es hacerlo mediante un control jerárquico de la red, o bien distribuyendo el control de la red entre agentes software que se encargan de gestionar tareas concretas. Precisamente, y como hemos explicado en el *Capítulo 6*, nuestra propuesta va en esta línea.

Algunas de las ventajas de los SMA son la escalabilidad y la tolerancia a fallos. Esto se deriva directamente del carácter distribuido, de la gestión adaptativa del tráfico realizado por los agentes. Como las redes suelen crecer, tanto en tamaño como en complejidad, el control distribuido y robusto pasa a jugar un papel primordial que es más importante cada día. Los cambios en la topología de la red pueden ser manipulados incorporando más agentes a la red. De este modo, cuando un nodo falla, como es nuestro caso en las congestiones, otro agente puede aprender a tomar el control para cambiar y reenrutar el tráfico nuevamente, o actuando activamente en la recuperación del tráfico hasta que el conmutador congestionado vuelve a su estado de normalidad.

En nuestro caso, la visión de sistema distribuido nos la aporta el SMA que empleamos para construir la arquitectura propuesta y también el propio protocolo TAP que engloba al SMA. Como en los SD [1], en el caso de los SMA podemos hablar de tiempo de respuesta, rendimiento, usuarios concurrentes o tareas paralelas. Y desde este punto de vista vamos a revisar algunas de esas propiedades de los SMA que tienen cierta relación con el concepto de SD. TAP es una arquitectura basada en un esquema de sistema multiagente y, como tal, podemos decir que se trata de una arquitectura distribuida en la topología de la VPN que la incorpora. De forma general TAP puede formar parte de un sistema de gestión de red[1] que aporte a la tecnología ATM todas las características de las que ésta carece y que han sido descritas en capítulos precedentes.

7.2. SISTEMAS Y ARQUITECTURAS DISTRIBUIDAS

Tradicionalmente se ha diferenciado entre cuatro niveles de arquitecturas para el control de problemas complejos: centralizados, jerárquicos, distribuidos e híbridos. Cada una de ellas tiene sus ventajas e inconvenientes, pero en el caso de las redes de telecomunicaciones podemos decir que la tendencia es que su control es claramente distribuido. Son muchas las definiciones de SD que pueden encontrarse en la literatura, y una de ellas en [2] indica que es una colección de computadores independientes que aportan a sus usuarios la sensación de trabajar en un único ordenador. Quizás la mejor forma de entender el concepto de SD es contrastarlo con los sistemas centralizados donde se depende de un elemento central que controla todo el sistema. Así, algunas de las ventajas de los SD respecto a los centralizados son: la velocidad, la fiabilidad, el crecimiento incremental, la distribución de las aplicaciones y los aspectos económicos. Otras propiedades importantes son: la compartición de datos, de servicios, las ventajas de comunicación entre diferentes componentes de un entorno y también la escalabilidad y la robustez ante posibles fallos del sistema. En cualquier caso, la resolución de problemas de forma distribuida aporta una gran potencialidad que permite incrementar la capacidad de cómputo y la disponibilidad de recursos de redes en aplicaciones locales y/o de mayor alcance.

Existen algunas cuestiones básicas para el diseño [1,2] y gestión [3] de SD y, entre ellas, destacamos las siguientes, que podemos ver como estrechamente relacionadas con algunos de los parámetros de QoS que ATM es capaz de garantizar:

- Transparencia: desde el punto de vista que los usuarios del SD puedan disponer de sus recursos, como si se encontrasen ante un sistema centralizado. Es decir, mientras se están disfrutando todas las ventajas de un sistema distribuido se tiene la impresión de disponer de las ventajas de un sistema centralizado. De este modo la característica de transparencia puede ser entendida desde diversos puntos de vista. Por un lado se habla de transparencia en cuanto a la localización física de los recursos que se usan. Transparencia en cuanto a la migración de los recursos, es decir, podemos mover geográficamente los recursos del SD sin afectarlo y manteniendo sus nombres. Transparencia en cuanto a la replicación de los recursos, que permite que el usuario del SD no necesite conocer cuántos recursos están replicados para aportar tolerancia a fallos al sistema. Transparencia respecto a la

[1] Tradicionalmente, se considera a los sistemas de gestión de red como una serie de herramientas para la monitorización y control de la red que suelen integrarse en atractivos entornos de usuario que permite la ejecución de comandos pensados para facilitar las labores de administración de la red.

concurrencia que permite que múltiples usuarios del SD puedan compartir conjuntamente sus recursos sin conocer este hecho. Por último, transparencia respecto al paralelismo que permite que las tareas realizadas en el SD se realicen en paralelo sin tenerse conocimiento de ello.

- Flexibilidad: En el caso de los Sistemas Operativos Distribuidos existen dos grandes corrientes en cuanto al diseño del SD. Por una lado está la corriente a favor de los sistemas monolíticos con un núcleo (kernel) que aporta la mayoría de los recursos, y por otro, la corriente partidaria de sistemas de microkernel que distribuyen las tareas y recursos entre varios equipos que disponen del menor número posible de recursos, pero sin perder de vista que todo el SD completo debe ofrecer todos los recursos de los sistemas separados. La posibilidad de elegir una corriente u otra, o los grados intermedios entre ambos extremos, es lo que podemos denominar como flexibilidad del SD.
- Fiabilidad: Este ha sido uno de los objetivos más importantes de los SD desde sus orígenes, y en nuestro caso particular es una de las aspiraciones principales. El planteamiento general justifica el uso de SD porque al distribuir los recursos y la carga, la fiabilidad del mismo es mucho superior que si se dispone de un sistema centralizado que se encargue de soportar todos los recursos y la carga. En el sistema centralizado cuando falla el nodo que lo soporta, todo deja de funcionar, mientras que si se trata de un SD la probabilidad de fallo se reparte entre los componentes del mismo.
- Funcionamiento: el rendimiento del SD debería ser superior al de un sistema centralizado para que su uso tenga justificación. En cierto modo, es cuestión de buscar el punto de equilibrio entre el rendimiento final del sistema y el resto de características comentadas. En algunos casos podría tener sentido sacrificar el rendimiento por la fiabilidad pero, en general, lo interesante es ser capaz de garantizar todas las características, entre las que el funcionamiento eficiente puede ser de las más importantes. En cuanto a la forma de medir el rendimiento pueden usarse varias técnicas como el tiempo de respuesta ante ciertos eventos, el índice de utilización del sistema, el número de accesos garantizados, el ancho de banda consumida en una red, el throughput final del sistema, etc. En nuestro caso una de las mayores aspiraciones está en conseguir una optimización del goodput de la red.
- Escalabilidad: esta propiedad estudiada en [4] debe permitir que el SD soporte un numero variable (generalmente elevado) de usuarios sin degradar el índice de rendimiento visto anteriormente. Esta es una característica que depende, tanto de aspectos software como hardware. En muchos casos, un recurso hardware (por ejemplo, un disco) puede acabar degradando el rendimiento final del sistema cuando el índice de simultaneidad de accesos de usuarios al disco crece por encima de un determinado valor. Sin embargo, también puede producirse el mismo problema de escalabilidad en aspectos software como podría ser un algoritmo o protocolo centralizado que atiende todas las peticiones de los usuarios que acceden al sistema.

Pero también aparecen desventajas en la utilización de SD como, por ejemplo, pérdida de seguridad y posibles problemas en la red. El diseño de SD debe conducir a características como la transparencia, flexibilidad pero, sobre todo, son atractivos para nosotros la fiabilidad, el rendimiento y la escalabilidad.

El carácter distribuido de los sistemas es posible gracias a la existencia de una red que permite la interconexión de todos los recursos del mismo. Por tanto, necesitamos nuevamente de elementos hardware y software para implementar la red que hace de nexo en el SD. Por un lado es necesaria la existencia de cableado y elementos de interconexión a la red y, por otro lado, es imprescindible la implementación de un protocolo de comunicación que permita, como poco: la especificación de la secuencia de mensajes que se van a intercambiar con la red y la especificación del formato de los datos de los mensajes intercambiados. Sabemos que un protocolo es implementado en un par de módulos software localizados en los dos (o más) extremos de una comunicación. Si además, los elementos de interconexión de la red que une los extremos de la comunicación implementan el protocolo podemos decir que éste está distribuido en la red. En el caso de ATM sabemos que se trata de una tecnología de conmutación de paquetes basada en un mecanismo de enrutamiento de paquetes conocida como *cell relay* que opera a mucha mayor velocidad que la conmutación de paquetes tradicional. Precisamente, se intenta conseguir esa velocidad de transmisión evitando que los nodos intermedios de la red se encarguen de las labores de control de flujo y de control de errores. Esto unido al carácter orientado a la conexión nos hace perder de vista el carácter distribuido de ATM. TAP aporta ese carácter distribuido a la red como un protocolo implementado en los extremos de la comunicación y en los conmutadores activos. Debemos destacar que la inclusión de TAP no resta rendimiento a la red, ya que el protocolo lo que hace es aprovechar los instantes de silencio de las fuentes para realizar las retransmisiones que sean solicitadas por las conexiones con requerimientos de GoS.

7.3. PROPIEDADES DE FUNCIONAMIENTO DE LOS SMA DISTRIBUIDOS

Según hemos visto en el *Capítulo 6*, un SMA está constituido por un grupo de agentes autónomos que interactuan unos con otros para alcanzar un objetivo común o individual. Para ello, los agentes deben coordinar sus acciones, para dar respuesta o soluciones como: asignar recursos; evitar o eliminar conflictos en la ejecución de acciones o por conflicto de intereses; conseguir eficiencia mejorando la predicción o reduciendo la redundancia. En líneas generales, lograr una eficiencia global del entorno en que se desarrollan que, en nuestro caso, es el ámbito de los sistemas de telecomunicaciones.

Pero en los SMA no sólo tienen importancia las características puramente funcionales[2] como la modelización del conocimiento, la movilidad, la inteligencia, la coordinación, etc. Debe prestarse también atención a otras propiedades no funcionales de los SMA, que son las relativas al funcionamiento[3] del propio SMA, como la escalabiliad, el rendimiento y la estabilidad. Estas características van recibiendo cada día mayor atención a medida que los SMA van madurando e implantándose [5]. Por esto vamos a centrar estos tres importantes conceptos.

- Tanto el rendimiento[4] o funcionamiento, como el throughput, el tiempo de respuesta y el número de usuarios concurrentes tienen también un papel importante en los SMA. Pero existen otros factores importantes que tienen costos y, por tanto, afectan al rendimiento. Podemos destacar los siguientes:
  - ✓ Coordinación de los protocolos usados, que suelen conllevar razonamiento y comunicación que llevan asociados costes computacional y de interconexión.
  - ✓ Modelo de conocimiento de los agentes que requiere de estructuras de datos complejas con costes de almacenamiento y de manipulación.
  - ✓ Modelo de racionalidad de los agentes que conllevan también un coste computacional debido a la evaluación y optimización del razonamiento.

  Por tanto, lo mismo que en los sistemas distribuidos, los indicadores de rendimiento en los SMA se expresan como costes computacionales y en throughput. El rendimiento es una función de la complejidad computacional de la implementación del SMA. En [5] se propone una definición de rendimiento en los SMA como medida en la que se usa un conjunto de indicadores estadísticos de las salidas del sistema y sus consumos de recursos donde algunos de ellos son el throughput, el tiempo de respuesta, el número de agentes o tareas concurrentes, el tiempo computacional y las sobrecargas debidas a las labores de comunicación.
- En cuanto a la escalabilidad, en el caso de los SMA, se refiere al modo en que incrementa la eficiencia de la función del sistema a medida que aumenta la complejidad de éste. Así, en la ingeniería del software distribuido el término escalabilidad se suele usar para referirse al incremento de la carga del entorno debido a un incremento en el número de componentes distribuidos. En el caso de los SMA el incremento en la carga del sistema es causada por la necesidad de mayor interacción a medida que crece el número de agentes que forman parte del SMA. La escalabilidad es por tanto una medida media de la degradación del rendimiento de los agentes individuales del sistema causada por el crecimiento del tamaño del sistema. Es decir, estamos ante un visión de complejidad algorítmica, ya que la escalabilidad del SMA depende, en el peor de los casos, del rendimiento del sistema que está limitado por una función polinómica de la carga del mismo.
- La estabilidad es una propiedad del equilibrio. Es decir, un sistema es estable si tras ser perturbado es capaz de volver voluntariamente a la situación de equilibrio original. En el caso de los SMA la estabilidad suele definirse en torno a las perturbaciones como el ruido, variación de los valores de ciertos parámetros del sistema, o las congestiones como es nuestro caso. El sistema será considerado inestable si después de un tiempo es incapaz de volver a la situación de equilibrio. En el caso de los SMA el punto de equilibro es difícil de definir, aunque se dice que están en equilibrio cuando las propiedades estadísticas de sus indicadores de rendimiento permanecen estacionarias para una variación dada en la carga externa del sistema. En nuestra arquitectura la situación de inestabilidad se produce cuando aparecen las congestiones en los conmutadores.

---

[2] Consideramos como propiedades funcionales aquellas que están relacionadas con la actividad vital y particular de cada agente individualmente o con el resto. Es decir, las funciones desempeñadas por cada agente del sistema.

[3] Definimos las propiedades de funcionamiento como aquellas que reflejan el grado de funcionamiento del SMA completo, que lógicamente viene determinado por el funcionamiento individual de cada uno de los agentes, o por la forma en que cada uno desempeña sus funciones.

[4] En el caso de los SMA el throughput suele expresarse como el número de transacciones por unidad de tiempo que el sistema es capaz de procesar.

Los sistemas expertos distribuidos son considerados como una de las grandes áreas de aplicación del procesamiento distribuido. Esta área es una de las que más nos interesa en nuestro caso, ya que las aplicaciones de sistemas expertos distribuidos destacan cuántos agentes negocian sobre soluciones colectivas. Uno de los objetivos de diseño de las aplicaciones distribuidas y de tiempo real ha sido el de reducir la frecuencia de comunicación entre agentes, o elevar la velocidad de comunicación de los recursos del SD. Sin embargo, las características de cada aplicación son las que definen el número de agentes que se necesitan, por lo que se requiere un cierto grado de comunicación entre los agentes. La literatura describe también paradigmas distribuidos y la referencia [6] presenta este paradigma diseñado por un sistema multiagente distribuido, donde cada agente en su entorno es capaz de computar autónomamente y cooperar con otros agentes para buscar una solución a un problema concreto.

Por otro lado, los agentes móviles también aportan atractivas características al procesamiento distribuido en un intento por evitar las debilidades de los sistemas centralizados como la falta de escalabilidad del control central, el incremento en los procesos y comunicaciones y su vulnerabilidad a los fallos en el sistema. Los procesos distribuidos necesitan comunicar entre sí para identificar y detectar eventos. De esta forma los agentes pueden aportar nuevas formas de enfrentarse a la problemática del control de los sistemas distribuidos para reducir su complejidad. Las referencias [7, 8] proponen la aplicación de los agentes móviles en el control de los SD, aunque podemos destacar que en realidad, la mayor parte de las investigaciones relativas a la aplicación de la tecnología de agentes a los sistemas de comunicaciones acaban reconociendo el carácter inherentemente distribuido de los resultados obtenidos.

7.4. PROTOCOLO TAP SOBRE UNA VPN DISTRIBUIDA

El concepto formal de VPN (Virtual Private Network) tiene una cierta similitud con las redes auto-contenidas, de forma que la VPN, además de ser virtual, supone que no está conectada a redes externas. Se supone que los datos son intercambiados de forma secreta entre los nodos. Además se supone que no hay conocimiento externo de la existencia de la red virtual. Cisco Systems [10] presenta su concepto de VPN indicando que se trata de "*un entorno de comunicaciones en el cual el acceso es controlado para permitir conexiones sólo dentro de una comunidad de interés, que es construido sobre alguna forma de partición de un medio de comunicación subordinado, medio que aporta servicios a la VPN de una forma no exclusiva*".

Existen diferentes definiciones de VPN para diferentes problemas y soluciones diferentes, por lo que nosotros deseamos relajar en cierto modo los anteriores conceptos de VPN y queremos quedarnos con la calificación de "discreta" en sus acepciones de separada, distinta o disjunta. No entramos por tanto en las características de independencia de redes exteriores, pero sí deseamos aclarar que nuestra propuesta de emplear TAP sobre una VPN pretende indicar que estamos ante una red privada o corporativa de tecnología ATM y que en su topología cuenta con conmutadores AcTMs que soportan la arquitectura y el protocolo TAP. De este modo, podemos considerar una VPN como una red privada construida dentro de una infraestructura de red pública, como sería en nuestro caso una red LAN, MAN o WAN de tecnología ATM.

La propuesta de esta VPN para el soporte de TAP coincide con las motivaciones generales de las VPN que se basan en criterios económicos, de privacidad, seguridad y, sobre todo, en aspectos de QoS. Al margen, hemos de destacar que, dadas las características comentadas hasta ahora y la justificación de SD que realizaremos en las siguientes secciones, justifican que TAP podría ser también usado en conexiones con requerimientos de GoS a través de troncales ATM. Quizás nuestra motivación más importante para proponer esta VPN es virtualizar o hacer invisible una parte de la red a usuarios externos de la VPN, mientras los usuarios internos pueden disfrutar de las ventajas que aporta TAP. No obstante, para enmarcar adecuadamente nuestros escenarios de trabajo nos restringimos a lo que denominamos VPN distribuida.

Mientras las velocidades de transmisión aumentan, los ACK acaban teniendo costes casi constantes entre los extremos por el tiempo *RTT* (R*ound-Trip Time*). Es decir, el ancho de banda en la red tiende a cero a medida que las fuentes incrementan sus requerimientos, pero el *RTT* se acerca asintóticamente a valores constantes que dependen del medio de transmisión que usemos. Las congestiones provocan el problema de los retardos, pero además, el control de flujo para acondicionar las prestaciones de emisor, receptor y red conduce también a desaprovechar los recursos de la red. Esto ocurre porque el ajuste provoca desaprovechar en muchos casos el ancho de banda disponible en la red que lleva hasta la posibilidad de que ésta quede desaprovechada estando inactiva en márgenes elevados del ancho de banda. Estas apreciaciones forman parte de las motivaciones generales de nuestra investigación, las cuáles van a ser descritas en *Capítulo 8*. Lo que podemos apreciar, como poco, es que las latencias debidas a los tiempos de *RTT* podrían ser repartidas a lo largo de toda la red como si de un sistema distribuido se tratase. Precisamente, la mejor forma de enfrentarse a esta aspiración es el desarrollo de nuevas arquitecturas y protocolos que permitan la distribución de carga, o

resolución de problemas de forma local, en lo que podríamos definir como sistemas distribuidos de área extensa y banda ancha.

La *Figura 7.1* presenta un escenario p-p en la VPN en el que se pueden observar tres nodos AcTMs que soportan la arquitectura TAP. En el nodo intermedio se produce una congestión del buffer que es detectada por el algoritmo EPDR. El identificador de la PDU que experimenta la congestión es usado para solicitar la retransmisión, la cual es enviada al nodo emisor que atenderá la retransmisión de esa PDU siempre que tenga suficiente tiempo de inactividad. De este modo, la retransmisión no es realizada desde el extremo de la comunicación, sino desde el nodo intermedio con lo que conceptualmente ahorraremos la mitad del tiempo *RTT*.

Del mismo modo, en el nodo final se produce una nueva congestión, por lo que se genera una nueva célula RM con el valor del índice de la PDU congestionada que es atendida por el nodo intermedio. Si este nodo intermedio tiene tiempo para atender la retransmisión y además la PDU está aún en la memoria DMTE, se encargará de reenviar nuevamente la PDU hasta el nodo final. De este modo ahorramos nuevamente la mitad del tiempo *RTT* ya que la retransmisión no llega al nodo emisor, sino que es atendida localmente al conmutador en que se produce la congestión. Además, en este caso se consigue evitar el posible problema de implosión sobre el nodo emisor que delega su labor de retransmisión en un nodo intermedio y siempre más cercano al nodo congestionado.

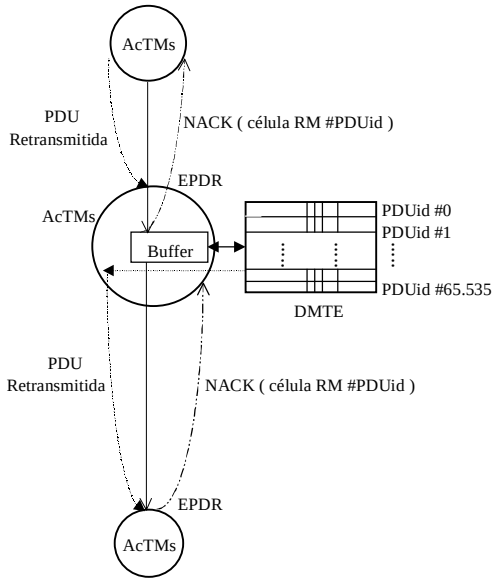

Figura 7.1. *Escenario p-p en VPN*

Podemos enfrentarnos a un nuevo escenario como el mostrado en la *Figura 7.2* en el que puede observarse una comunicación p-mp, donde además se representa la posibilidad de una VPN con nodos intermedios entre los destinatarios de la conexión y el nodo activo que se encarga de atender las retransmisiones antes de que lleguen a la fuente de tráfico. Como puede observarse en la *Figura 7.2*, los nodos no activos de la VPN que no soportan el protocolo activo TAP sólo van a atender los NACK como células RM nativas, por lo que la única labor que realizan estos conmutadores es la de reencaminar las peticiones de retransmisión en sentido al emisor. Si antes de llegar al emisor existe en la VPN un nodo AcTMs que aún contiene en su DMTE la PDU solicitada y tiene suficiente tiempo para atender la solicitud, se encargará de acceder a la memoria DMTE a través de los agentes software y reenviar la PDU nuevamente en el sentido al destinatario que la solicitó. En este caso también podemos ver cómo el ahorro de tiempo *RTT* puede ser importante y, sobre todo, al tratarse de una transmisión multipunto el problema de implosión será mucho mayor que en el caso del escenario de la *Figura 7.1*.

El último escenario al que nos podemos enfrentar es al representado en la *Figura 7.3*, en el que aparecen múltiples fuentes emisoras y múltiples destinatarios del tráfico. En realidad, el escenario mp-mp consiste en la yustaposición de *n* conexiones p-mp. En este caso el riesgo de implosión en los emisores es también evidente, por lo que la necesidad de disponer de un conmutador activo en el que los emisores puedan delegar las tareas de retransmisión es una opción interesante. La *Figura 7.3* introduce la posibilidad que entre el nodo activo intermedio y las fuentes de tráfico puedan existir en la red un número indeterminado de conmutadores no activos.

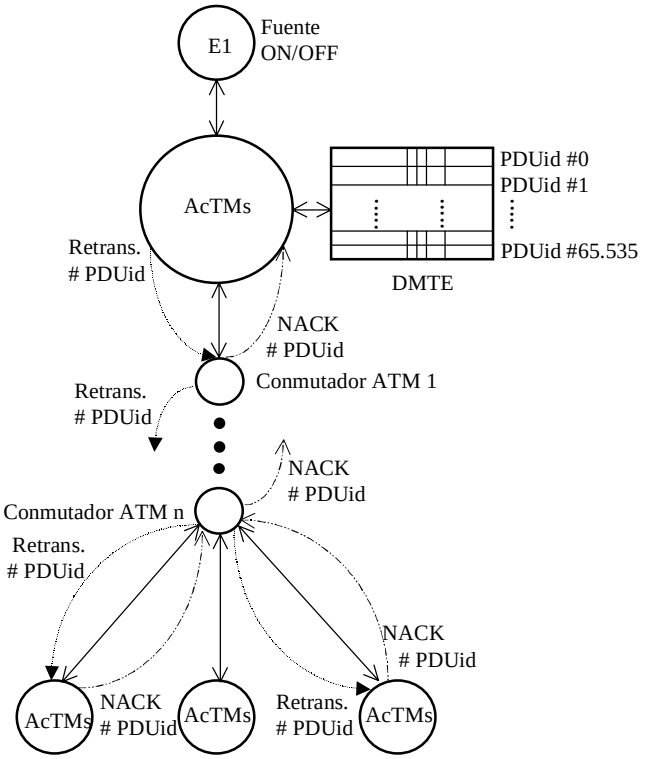

Figura 7.2. *VPN con un escenario p-mp y nodos no activos*

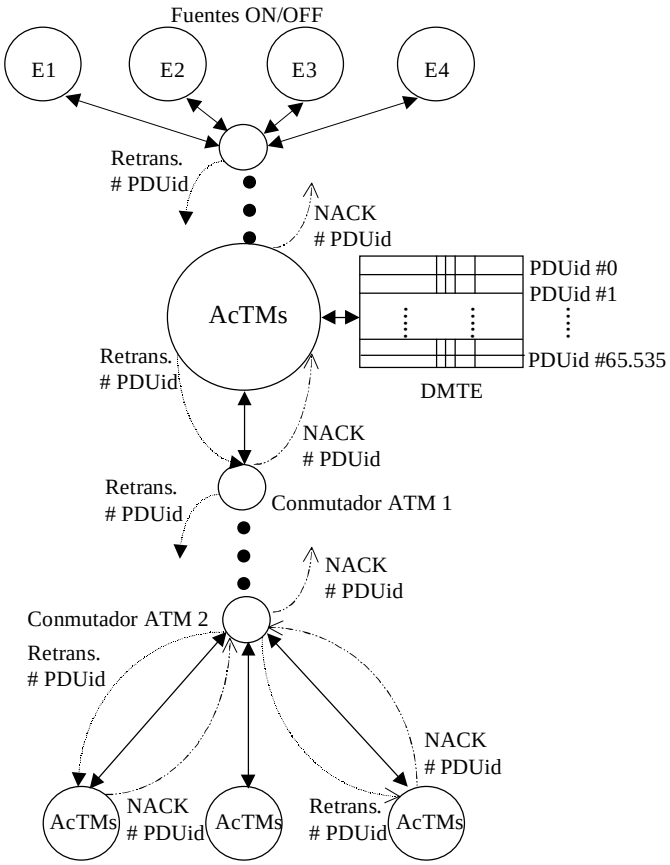

Figura 7.3. *VPN con escenario mp-mp*

A la vista de los tres escenarios comentados puede entenderse que el protocolo TAP está funcionando sobre toda la topología de la red, de forma que, por el hecho de disponer de un software distribuido (como el mismo protocolo y los agentes software) y un conjunto de escasos recursos hardware compartidos (como los propios conmutadores AcTMs y las memorias activas de cada uno de ellos), podemos decir que estamos ante un sistema distribuido que soporta el protocolo activo y distribuido TAP.

Podemos decir que el protocolo TAP cumple con todos los requerimientos de los algoritmos descentralizados propuestos en [2] que dan lugar a algoritmos distribuidos:

- Ningún componente de la VPN gestiona información completa sobre el estado completo del sistema. Es decir, cada conmutador activo almacena las PDU que le permite su propia DMTE, y no existe la necesidad de uso de ningún tipo de base de datos del sistema que pueda perjudicar el rendimiento final del mismo.
- Los nodos del sistema toman decisiones basándose únicamente en su información local. En este caso este aspecto es de crucial importancia para nosotros, ya que el sistema multiagente juega un importante papel en el sistema distribuido. Los agentes software locales a cada AcTMs aplican su propia inteligencia para actuar en su propio conmutador, aunque pueden aplicar la posibilidad de solicitar las retransmisiones a otros nodos hasta llegar el emisor.
- El fallo de uno de los componentes del sistema no pone en peligro la integridad del algoritmo TAP ya que la búsqueda de PDU congestionadas son locales a la congestión, pero pueden irradiarse en sentido a la fuente y a partir del nodo congestionado. Debemos destacar que ésta ha sido una de las precondiciones en nuestras investigaciones, ya que no podemos obligar a que todos los conmutadores de la red tengan que soportar el protocolo. En las figuras hemos comprobado cómo los nodos que no soportan TAP serán capaces de procesar las células RM nativas y estándares hasta llegar al emisor de la conexión. Si no hubiésemos previsto esta posibilidad nos encontraríamos con el inconveniente de poder asumir el coste económico de nuestra propuesta, porque obligaría a cambiar toda la equipación existente en una red para poder disponer de la GoS que ofrece TAP. Además, este hecho también impediría cumplir los requerimientos de escalabilidad, robustez, fiabilidad, disponibilidad, etc. de los sistemas distribuidos.
- No existe una asunción implícita de que exista un reloj global de todo el sistema. Aunque este requerimiento es más laxo en los sistemas distribuidos, en nuestro caso podemos garantizar que el protocolo no tiene ninguna necesidad de una sincronización de tiempos en el sistema. Los flujos dentro del sistema pueden seguir sus propios ritmos, sin poner en peligro el funcionamiento del algoritmo. En algunos casos puede ocurrir que las memorias DMTE no dispongan de las PDU solicitadas porque una fuente genere un tráfico demasiado fluido, pero es preferible asumir este riesgo antes que afectar al funcionamiento completo del sistema. Podemos destacar que en nuestro caso, el único mecanismo de control general en el sistema es la numeración secuencial de las PDU desde que salen del emisor. De este modo disponemos de una “ventana” de 65.535 PDU que pueden ser retransmitidas hasta que vuelve a comenzar un nuevo ciclo de PDU.

Sabemos ya que el principal objetivo de la arquitectura TAP es la de proporcionar soporte a un protocolo activo que aporta garantía de servicio a las conexiones de una red ATM. Para ello incorpora agentes software y un conjunto de refinados mecanismos de la ingeniería de protocolos con la intención de eliminar los problemas que ya han sido identificados como la implosión, la fragmentación de paquetes y el interleaving. De este modo, las propiedades principales de la arquitectura TAP, relacionadas con su carácter distribuido son:

- El carácter nativo ATM, que permite su completa integración con toda la tecnología ATM existente, ya que el protocolo es soportado sólo por los conmutadores AcTMs que pueden estar distribuidos por la topología de la VPN. Esto permite que no sea necesario cambiar todos los elementos de la red, sino sólo aquellos que se consideren oportunos y, sobre todo, aquellos conmutadores que se conozcan previamente como fuente potencial de congestiones.
- Eficiencia, debido a que TAP permite optimizar el rendimiento de la red consiguiendo que el goodput que suele degradarse en las situaciones de congestión sea mantenido en niveles mucho superiores a los que se obtendrían en el caso de no aplicar TAP. Además, debido a que se emplean los periodos de inactividad de las fuentes para recuperar las congestiones, se garantiza que el throughput general de la red no se vea comprometido ya que sólo se realizan retransmisiones cuando la red dispone de suficiente tiempo de silencio para no afectar a las fuentes de tráfico que puedan estar activas en cada instante de tiempo.

- Escalabilidad, ya que la complejidad y el rendimiento del sistema no se incrementan a medida que se incrementa el número de elementos que forman parte de nuestra arquitectura distribuida. Es decir, la incorporación de nuevos conmutadores AcTMs a la red no hace peligrar el rendimiento de la red, si no que se consigue aumentar el grado de GoS de las conexiones privilegiadas, o bien aumentar el número de conexiones privilegiadas que puede emplear el sistema.
- Justicia, pues los esquemas introducidos de colas justas permiten caracterizar el tráfico, consiguiendo aportar mayores anchos de banda a las conexiones que lo solicitan, y también aportando GoS a las conexiones que establecen su conexión como garantizadas o privilegiadas. No obstante, el protocolo implementado no descuida las situaciones de injusticia que puedan producirse, evitando que las conexione con menos necesidades puedan verse relegadas.
- Robustez, en el sentido que el SMA propuesto aporta robustez por sí mismo. Es decir, en situaciones en que algún agente del sistema deje de funcionar o, en el peor de los casos el SMA completo no actúe, no provocará ningún problema a la red. Únicamente dejarán de garantizarse todos los aspectos que TAP cuida, pero el funcionamiento general de la VPN no se ve amenazado por fallos de TAP.
- La simplicidad de la arquitectura es también otra de las ventajas de TAP. El SMA propuesto incorpora un total de cinco agentes software de los cuáles dos son programables y dinámicos en lo que respecta a su detección de eventos en tiempo real y durante la explotación de la red. Además, el esquema general de comunicación entre los agentes es también muy sencillo sin incorporar excesiva complejidad algorítmica, ni coste computacional por el acceso a complejas estructuras de datos, ni tampoco sobrecargar en las labores de coordinación entre los diferentes agentes del sistema.
- Estabilidad o equilibrio, ya que las simulaciones realizadas permiten demostrar que, ante la aparición de ruidos en el sistema (en nuestro caso, congestiones provocadas por tráfico de backgroud), éste es capaz de actuar poniendo en marcha el mecanismo de recuperación de PDU y, una vez, abandonada la situación de congestión el sistema es capaz de volver o converger a su punto de equilibrio o estabilidad como si de un sistema dinámico se tratase.
- Economía, ya que la equipación hardware que hay que incorporar a los conmutadores para que soporten el protocolo propuesto no va más allá de una simple memoria de almacenamiento de PDU. Además, se propone también la implementación de la mayor parte de los esquemas software desarrollados mediante hardware para conseguir un mayor grado de integración y funcionamiento.

Para conseguir todas estas características se requiere de un sistema claramente distribuido, en el que no sea necesaria ninguna visión ni control central de la red ni de sus datos. No obstante, como lo que se persigue es un óptimo funcionamiento de la red que incorpora TAP es importante destacar que ésta permite al operador de la red su intervención para ajustar los parámetros de las conexiones en las situaciones en las que se puede perder el estado de equilibrio del sistema. Es decir, aunque TAP es capaz de resolver el problema de las pérdidas de PDU, el SMA permite al operador de la red la intervención (introduciendo órdenes y objetivos) para intentar evitar las situaciones de congestión generando acciones de los agentes programables del sistema.

7.5. SISTEMAS DISTRIBUIDOS FIABLES CON CORBA

CORBA (Common Object Request Broker Architecture), se ha convertido en el estándar para diseñar sistemas abiertos distribuidos [11], también en entornos de comunicaciones. En sus inicios CORBA estuvo principalmente pensado para la distribución de información en conexiones p-p y no ofrecía soporte para el desarrollo de aplicaciones fiables con comportamiento predecible en sistemas distribuidos. Sin embargo, no han tardado en aparecer extensiones como [9,12] pensadas para que aplicaciones más sofisticadas como teleconferencia, VoD y otros servicios para grupos de usuarios soporten fiabilidad y tolerancia a fallos.

La Programación Distribuida Orientada a Objetos (PDOO) puede verse como una variante del modelo cliente-servidor, y es destacable que el diseño orientado a objetos y la implementación de sistemas distribuidos aportan importantes ventajas como la separación del servicio garantizado de los objetos de la propia tecnología de implementación. Sin embargo, el soporte de aplicaciones distribuidas fiables supone importantes esfuerzos ante lo impredecible de los retardos en las comunicaciones y de los fallos parciales en las redes como pueden ser los problemas debidos a las congestiones que estudiamos en esta tesis.

La fiabilidad en CORBA ha sido conseguida con el desarrollo de abstracciones para la gestión de grupos multicast que han complicado de forma importante los sistemas distribuidos. La fiabilidad de los grupos multicast requiere que las solicitudes sean confirmadas por todos los miembros de los grupos. Por esto los

posible fallos pueden conducir al bloqueo de algunos clientes que esperaran indefinidamente la respuesta de miembros de los grupos que están experimentando problemas. Para solventar estas situaciones se han propuesto soluciones como las solicitudes atómicas[5], que permiten mantener la consistencia de los grupos multicast cuando se produce la detección y propagación de fallos.

7.6. CONCLUSIONES

En este capítulo hemos querido destacar las principales características de los SD con la intención de incorporar las ventajas de la computación distribuida a nuestras investigaciones. Hemos podido ver cómo la propia definición del concepto de SD implica aspectos hardware y software que, en nuestro caso concreto, se resuelven con la propuesta de la arquitectura TAP sobre conmutadores activos AcTMs. Los conmutadores activos constituyen una VPN distribuida (por el carácter distribuido del protocolo TAP) y, por otro lado, contamos con la equipación en esta arquitectura de las técnicas activas que aportan los agentes software del SMA-TAP. Además, todo el sistema distribuido resultante es controlado mediante el protocolo TAP visto como un algoritmo descentralizado. Toda esta propuesta nos permite argumentar que TAP constituye un SD que es capaz de garantizar características propias de los SD como la escalabilidad, el rendimiento, la fiabilidad, la flexibilidad, la robustez y la transparencia para los usuarios que requieran las conexiones garantizadas en la VPN propuesta. Por otro lado, CORBA aparece como la forma de incorporar la distribución de servicios de una forma estándar que soporta además la tecnología de los agentes software. Aunque nuestras implementaciones no usan CORBA, hemos identificado esta posibilidad como una futura línea de trabajo que permita la mejora de TAP.

REFERENCIAS

[1] George Coulouris, Jean Dollimore, and Tim Kindberg, "Distributed systems: concepts and design," *Ed. Addison-Wesley* (1994).

[2] Andrew S. Tanenbaum, "Distributed Operating Systems," *Ed. Prentice Hall International Editions*, (1995).

[3] William Stallings, "SNMP, SNMPv2, SNMPv3 and RMON 1 and 2," *Ed. Addison-Wesley*, (1999).

[4] Pere Vilà and Josep L. Marzo, "Scalability Study and Distributed Simulations of an ATM Network Management System based on Intelligent Agents," *SPECTS'2K 2000 SCS Symposium on Performance Evaluation*, (2000).

[5] L.C. Lee, H. S. Nwana, D. T. Ndumu and P. De Wilde, "The stability, scalability and performance of multi-agent systems," *http://www.labs.bt.com/projects/agents/publish/papers.htm*

[6] Jiann-Liang Chen, Shingfeng Lin, Ming-Yi Lee, "Distributed fault management over ATM networks," *Proceedings of the Sixth IEEE Computer Society Workshop on Future Trends of Distributed Computing Systems*, pp.196-201. (Oct. 1997).

[7] Simon Steward and Steve Appleby, "Mobile software Agents for Control of Distributed Systems Based on Principles of Social Insect Behaviour," *Proceedings Conference Singapore ICCS'94*, pp. 549-553, (1994).

[8] Gwan Joong Kim, Young Sun Kim and Hyeong Ho Lee, " A design of Management System for ATM Switches Using Mobile Agent Concept," *Global Telecommunications Conference, GLOBECOM 1998. The Bridge to Global Integration. IEEE*, pp. 1694-1698 vol.3, (1998).

[9] Robert Orfali, and Dan Harkey, "Client/Server Programming with Java and CORBA," *Ed. John Wiley and Sons*, (1998).

[10] Paul Ferguson and Geoff Huston, "White paper: What is a VPN " *http://www.employees.org:80/~ferguson/* (2000).

[11] Object Management Group, "The Common Object Request Broker: Architecture and Specification," (1995).

[12] Sean Landis, and Silvano Maffeis, "Building Reliable Distributed systems with CORBA," *http://citeseer.nj.nec.com/landis97building.html*, (1997).

[5] La atomicidad en las solicitudes garantiza que las solicitudes enviadas a los grupos de objetos serán recibidas por todos los miembros del grupo o por ninguno.