ANEXOS

ANEXO I. PATHRATE

Pathrate es una herramienta de estimación del ancho de banda basada en la medición de la dispersión de pares y trenes de paquetes. Se fundamenta en la teoría del *Packet Pair.*

En general, las estimaciones de ancho de banda realizadas mediante el método del *Packet Pair* siguen una distribución multimodal; es decir, las medidas se agrupan alrededor de ciertos valores cuya probabilidad de aparición es mayor (ver figura A.1). Sucede así debido a la presencia de tráfico interferente en la red. La capacidad del *path*, o lo que es lo mismo del *narrow link*, será uno de esos modos.

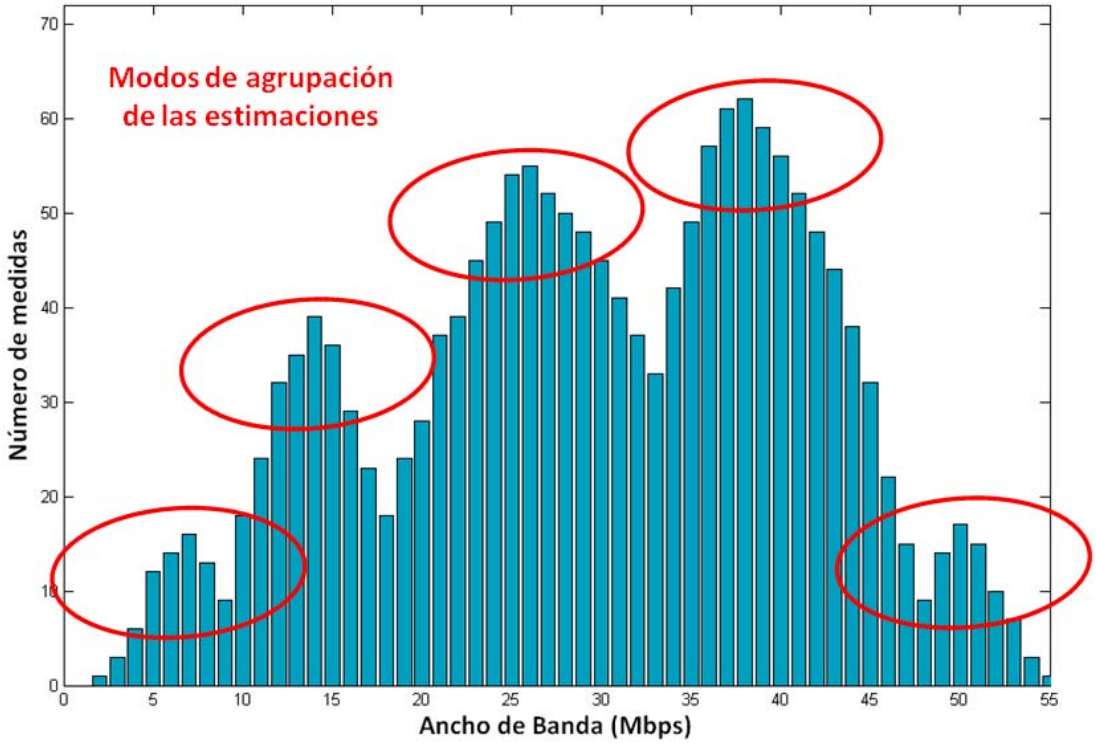

Figura A.1. Distribución multimodal de las estimaciones de capacidad realizadas mediante *Packet Pair*.

Pathrate está basado en el envío de ráfagas de diferentes longitudes conteniendo paquetes de tamaños diversos, mediante el cual se logra el debilitamiento de los modos ocasionados por el *cross traffic,* de manera que el modo que prevalezca sea el asociado a la capacidad del *path*.

I.1 Fases

Consta de tres fases; un esquema de funcionamiento sería el mostrado en la figura A.2:

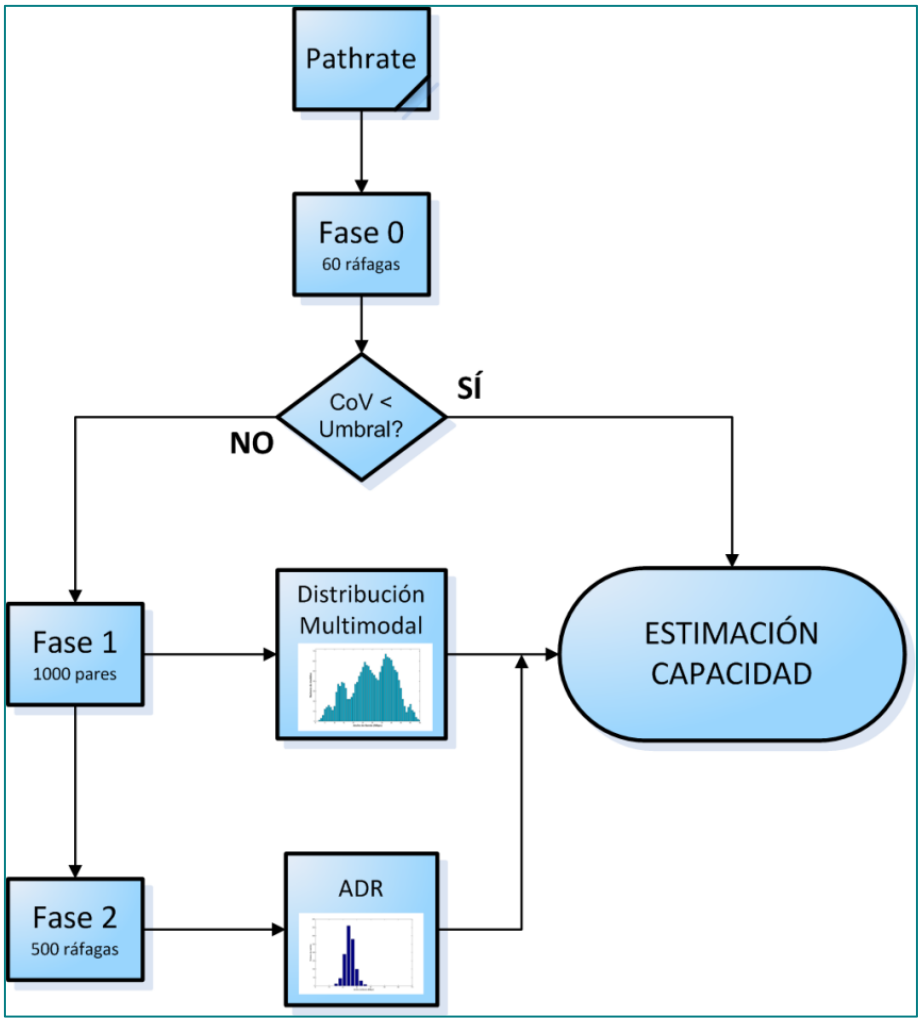

Figura A.2. Esquema de funcionamiento de *Pathrate.*

FASE 0

El objetivo de la fase inicial es, en caso de que haya poca congestión en la red, obtener el ancho de banda del *narrow link* en poco tiempo sin enviar apenas información. Si no es posible se llevan a cabo las dos siguientes fases, con su consiguiente envío de una cantidad considerable de información.

Se produce el envío de 60 ráfagas de paquetes cuya longitud va desde los 2 hasta los 10 paquetes, cuyo tamaño es el máximo que la red permite transferir sin fragmentación. En este PFC no se busca estimar la tasa de pérdidas del *path*, por lo que se considera una red ideal sin pérdidas y el tamaño máximo de paquete para *Pathrate* será el MTU, 1500 Bytes.

Aplicando el algoritmo del *Packet Pair* se obtienen 60 medidas de Capacidad. Si el coeficiente de variación (**CoV**, **C**oefficient **o**f **V**ariation) de las estimaciones

preliminares es menor que un cierto umbral, la capacidad resultado será el promedio del 90% de las medidas, tras eliminar las más pequeñas y grandes (el 10% del total). En caso contrario será necesario llevar a cabo las otras dos fases.

FASE 1

En esta fase *Pathrate* trata de descubrir los diferentes modos de agrupación de las estimaciones. Para ello enviará 50 pares de paquetes de cada tamaño, comenzando en 550 Bytes (tamaño mínimo para *Pathrate*) y llegando hasta el tamaño máximo de la herramienta, 1500 Bytes. Un histograma de las estimaciones da como resultado una gráfica como la representada anteriormente en la figura A.1.

FASE 2

Con el fin de desenmascarar el modo de la capacidad del resto se procede a introducir en la red 500 ráfagas de 100 paquetes cada una, todos ellos del tamaño máximo. La estimación de capacidad más frecuente será el **ADR** (**A**verage **D**ispersion **R**ate). Este valor delimita inferiormente la capacidad, de manera que la capacidad será aquel modo de aparición más frecuente de entre los que tienen un valor superior al ADR.

I.2 Umbral

Si el CoV de la estimación de la capacidad llevada a cabo en la "Fase 0" no supera el umbral, las fases 1 y 2 no se llevan a cabo, con el consiguiente ahorro de ancho de banda y recursos en la red. Este umbral se establece a partir de la experiencia, y en función de su valor el método opta por ser más preciso o menos intrusivo. Existe un compromiso:

- Si el umbral se escoge poco exigente el método se *"conformará"* en más ocasiones con la estimación previa. En estos casos la medida puede ser menos precisa, pero a su vez se evita el congestionamiento de la red.

- Si el umbral es muy exigente, rara vez será suficiente la fase inicial (salvo situaciones en que apenas exista *cross traffic*). El resultado podrá ser más preciso a costa de una mayor congestión de la red.

I.3 Intrusividad

Se ha observado que una mayor o menor intrusividad es regulada asimismo mediante el umbral. En caso de ser necesaria la realización de las tres fases, la cantidad de información enviada por *Pathrate* será:

Fase 0	4.01 Mb
Fase 1	15.64 Mb
Fase 2	572.20Mb
Total	591.85Mb

Es una cantidad considerablemente alta teniendo en cuenta que no es información útil como tal, sino que su utilidad radica en la estimación la capacidad de un *path,* que posiblemente esté siendo congestionado o interferido severamente por la herramienta.

I.4 Tiempo de estimación

Al igual que sucede con la intrusividad, el tiempo necesario para la estimación de la capacidad dependerá del umbral escogido. Si se llevan a cabo las fases 1 y 2 se requerirá un tiempo mayor que en el caso contrario.

Por otra parte, el intervalo temporal entre dos ráfagas consecutivas no está establecido. El criterio seguido en este caso es el siguiente: en caso de ser necesaria la ejecución del método completo no debe saturarse la línea por completo, de tal manera que los usuarios continúen empleando la red sin que *Pathrate* la congestione. Por ejemplo, no sería viable el envío de 600 Mb a través de un modem de 56 Kbps, al igual que enviar 600 Mb a través de una red de acceso a 10Mbps supondrá un tiempo elevado (siendo superior en el caso de no permitir una intrusividad total). La saturación del enlace dependerá de su capacidad. En este PFC se ha decidido que una realización completa de *Pathrate* dure aproximadamente 70 segundos. No es un valor excesivamente grande tratándose la capacidad de un parámetro tan poco cambiante en una comunicación establecida en la red.

Es necesaria una pequeña matización: las oscilaciones en la estimación de la capacidad llevada a cabo por *Pathrate* y que varían en torno al valor de capacidad real son consecuencia del empleo de un histograma como elemento discriminador. Las celdas del histograma tienen una cierta anchura, causa de ese error.

ANEXO II. OPNET

Se han diseñado unas herramientas de monitorización de capacidad junto con unas pruebas para ponerlas en práctica, y se ha llevado a cabo con el apoyo de la herramienta OPNET. El punto principal era estimar la capacidad; se han implementado los elementos necesarios para llevar a cabo las medidas.

OPNET MODELER permite crear estos componentes siguiendo un esquema fijado de programación. Los pasos necesarios para implementar cada uno de ellos son similares en todos los casos, aunque el bloque funcional que representa las tareas que se deben realizar varía en función de las necesidades de cada dispositivo.

Un proyecto en OPNET es jerárquico: consta de diferentes niveles, todos entrelazados. En cada uno de ellos se dota al diseño de unas especificaciones y posibilidades. En el nivel superior se encuentra el *Project Editor*, que engloba al resto de niveles y permite llevar a cabo simulaciones. Bajo este nivel, en el que encontramos el nivel de red, podemos hallar subredes, nodos y enlaces. Existe una gran diversidad de modelos de cada uno de estos componentes, aportando diferentes capacidades y características básicas. Un nodo queda definido por su *process model*, que determina sus funciones y su comportamiento.

Antes de comenzar el diseño de un proyecto es necesario tener claros qué elementos van a tomar parte (principalmente nodos y enlaces) y cuáles van a ser sus tareas a realizar. OPNET aporta una enorme variedad de herramientas, opciones de diseño y funcionalidades que se escapan del objetivo de este PFC. Únicamente se comentan en este anexo aquellas que han sido empleadas a lo largo de este trabajo.

A continuación se presenta el esquema general de un proyecto de OPNET:

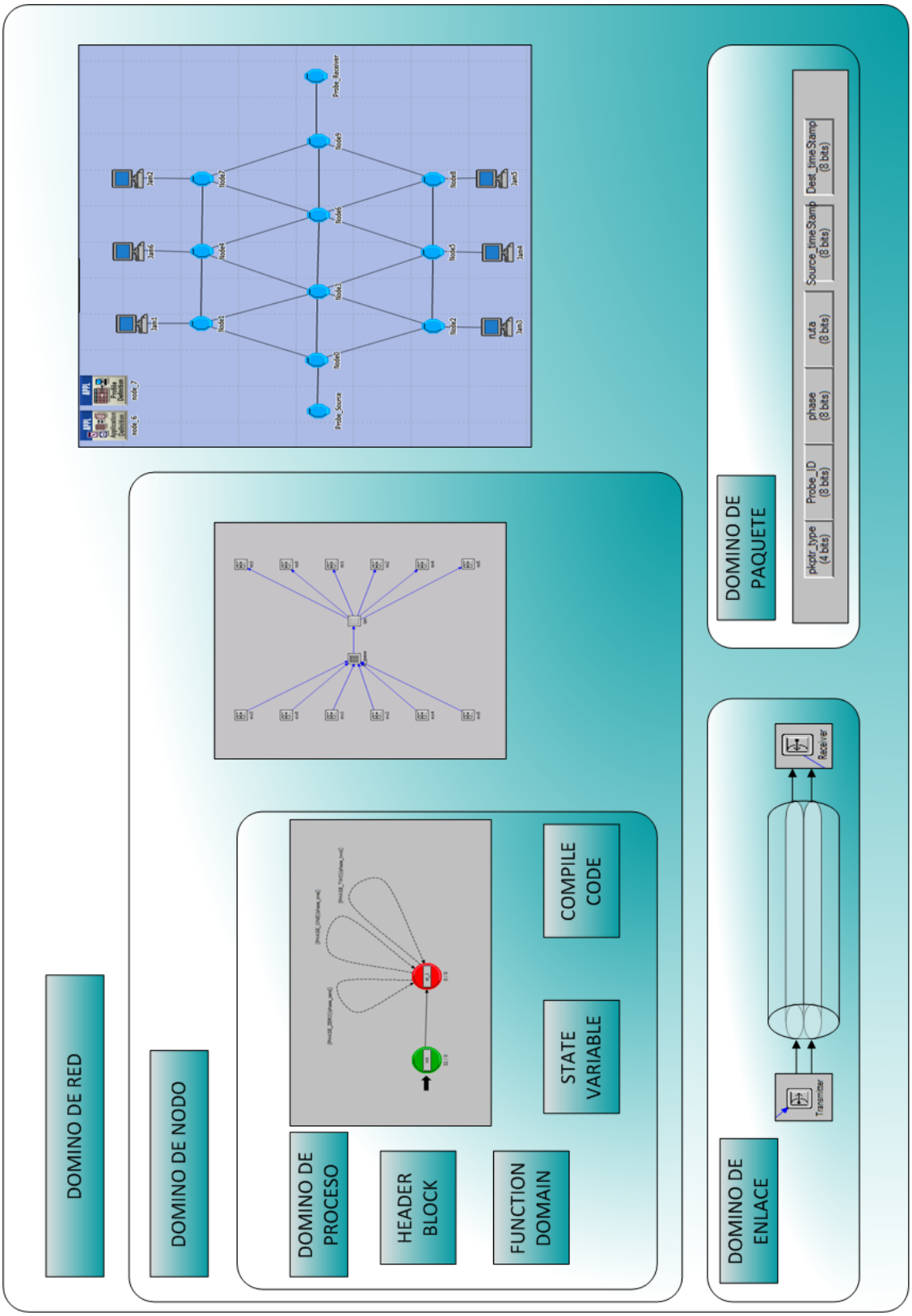

Figura A.3. Relación jerárquica de niveles en OPNET Modeler.

Siguiendo el esquema de programación citado anteriormente, el primer paso una vez iniciado el proyecto es determinar los nodos y enlaces que componen la red.

Dominio de Nodo

Este nivel proporciona dispositivos a interconectar en el nivel de red, que en términos de OPNET se llaman nodos. Se corresponderían con equipos de comunicaciones reales, tales como routers, terminales, etc. Son desarrollados en el *Node Editor* y se componen de bloques más pequeños llamados módulos **[31]**.

En este PFC se han utilizado tres tipos de módulos:

- **Transceptores** (*transceivers*): En este término se incluyen transmisores y receptores. Un nodo debe poseer el mismo número de los primeros que de los segundos. Permiten al nodo conectarse mediante enlaces en el nivel de red. Deben configurarse para una cierta tasa máxima y formatos de paquete aceptados. Cada transceptor puede tener uno o más canales que a su vez pueden configurarse independientemente. En la figura A.4 se observa la correspondencia de canales.

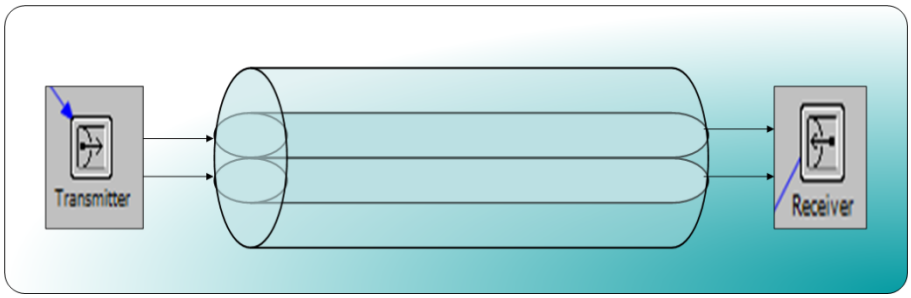

Figura A.4. Correspondencia entre transmisor, receptor y canales.

En este PFC sólo se ha utilizado un canal por enlace, aunque en algunas ocasiones se hayan configurado transceptores de dos canales.

- **Procesadores**. Estos módulos representan la cabeza pensante del nodo. Su comportamiento se especifica completamente en el *Process Model*, que especifica las funciones que se deben ejecutar. Se interconectan con otros módulos mediante los *packet stream*.

- **Colas**. Al igual que los procesadores, pueden ejecutar *process model* que describan el comportamiento de un procedimiento o protocolo en particular y se conectan con otros módulos mediante *packet stream*, permitiéndoles enviar y recibir paquetes.

La conexión de transceptores y colas con los procesadores es un paso importante en el diseño del nodo. En función de cómo se realicen, el *process model* puede variar su modo de actuar ante la recepción de una interrupción. El *Node Model* de un *router* empleado en el Escenario 2 se muestra en la figura siguiente; se pueden observar los elementos nombrados anteriormente: procesador, cola, emisores, receptores y *packet stream*.

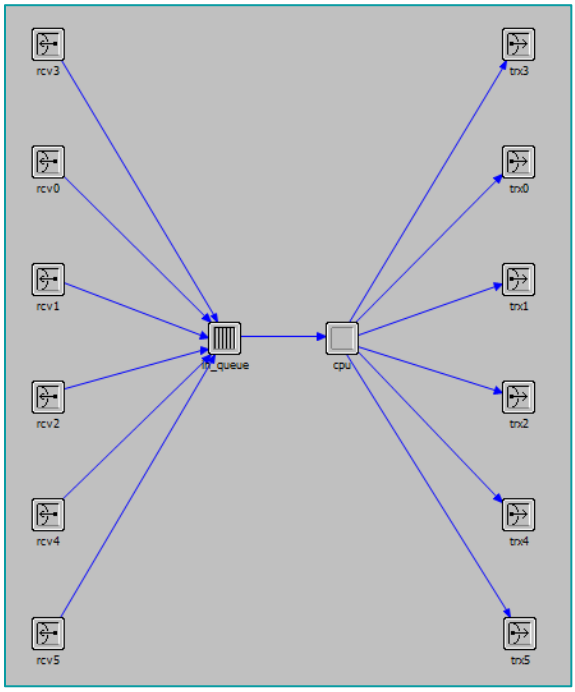

Figura A.5. Ejemplo de *Node Model*.

Dominio de Proceso

Como se ha comentado, una red en OPNET está compuesta de nodos individuales, cada uno formado por módulos. El *process model* define el comportamiento de cada módulo. En la figura A.6 se muestra el process model configurado en uno de los *router* diseñados en este PFC.

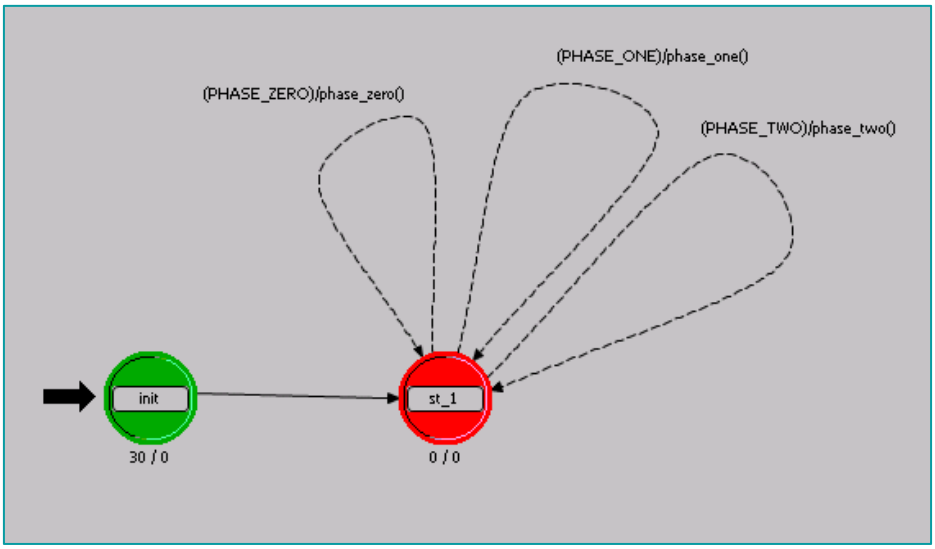

Figura A.6. ***Process Model*** **de un nodo** ***router*****.**

Un *process model* es una máquina de estados finita (FSM). Representa la lógica y el comportamiento de un módulo. OPNET emplea la FSM para implementar el comportamiento de un módulo. Las FSM emplean estados y transiciones para determinar qué acciones realiza el módulo en respuesta a un evento. El estado es la condición de un módulo y una transición es un cambio de estado en respuesta a un evento. Es posible emplear una FSM para controlar el comportamiento de un módulo puesto que OPNET permite al diseñador incluir fragmentos de código en C o C++ en estados y transiciones. Estos fragmentos se denominan *executives* y son de varios tipos: *enter executive* (se ejecuta cuando el modulo se pasa a un estado), *transition executive* (en respuesta a un evento específico) y *exit executive* (ejecutada al salir de un estado).

Una vez definidos los estados y transiciones se debe programar el *process model*. Se expresan en Proto-C, lenguaje que combina diagramas gráficos de estado-transición, código C/C++ y una librería de *Kernel Procedures* que proveen de funcionalidades de necesidad común. En el diseño del *process model* deben ser programados diferentes campos y elementos. En primer lugar se debe rellenar el *Header Block*; en él se definirán, entre otros, los *packet stream* y las transiciones que se encargarán de servir cada evento generado a través de esos flujos de entrada. Un ejemplo de HB empleado por la herramienta *Pathrate* se muestra en la figura siguiente:

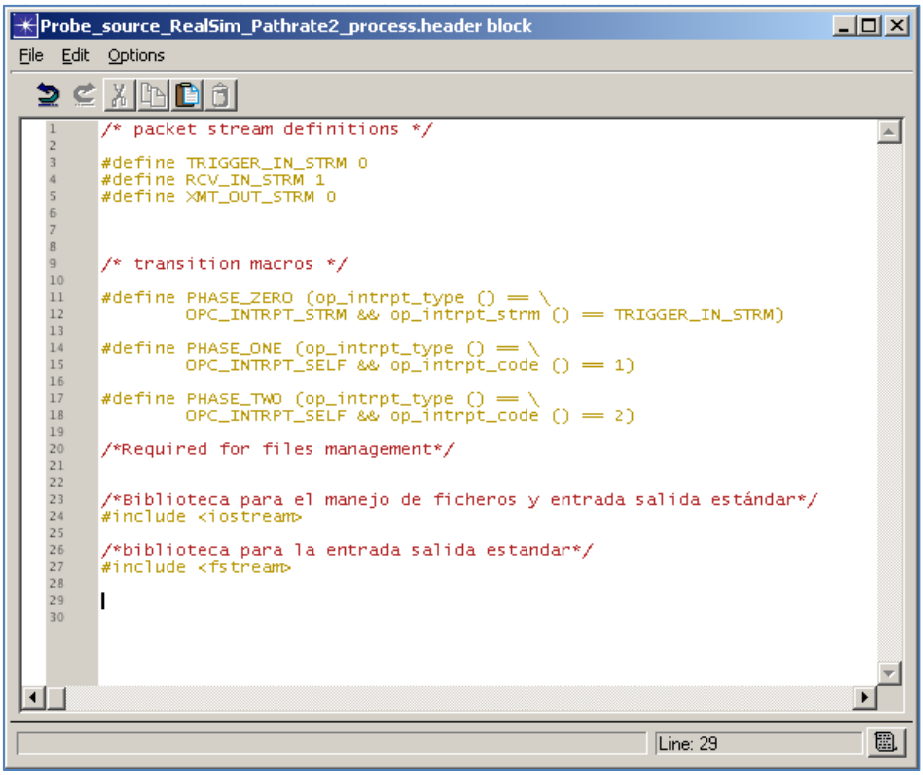

Figura A.7. Ejemplo de *Header Block* empleado en el *process model* del nodo fuente de la herramienta *Pathrate*.

A continuación se define el último bloque que modelará el comportamiento del nodo, el *Function Block* (FB). EL FB permitirá definir una función a ejecutar por un proceso. La tarea fundamental del programador es desarrollar este *Function Block*. Un ejemplo de *Function Block* empleado en uno de los *router* de los escenarios se muestra en la figura A.8:

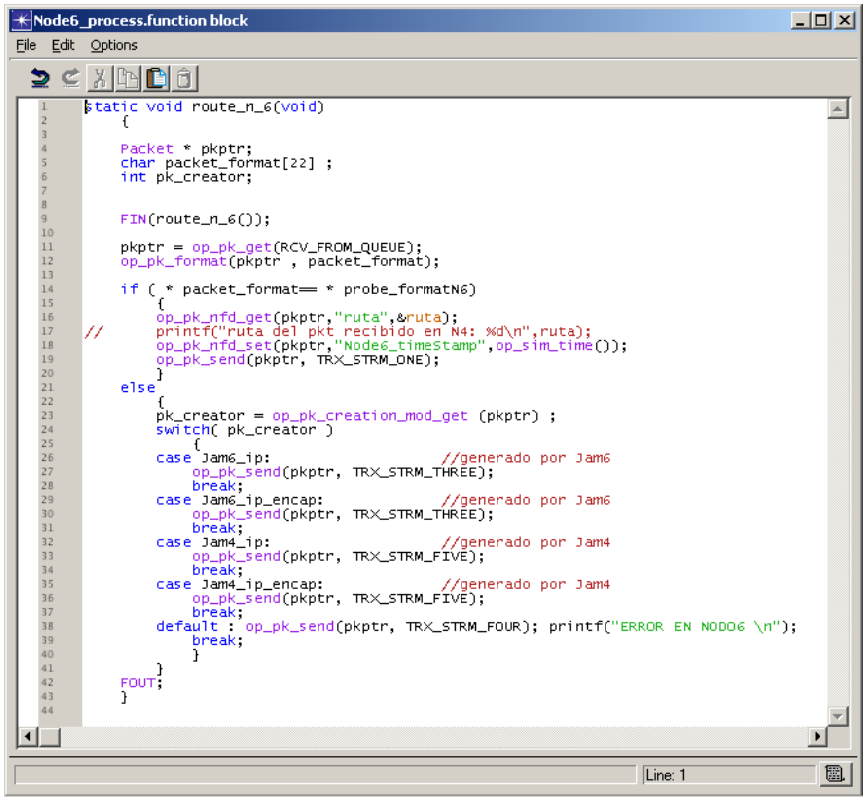

Figura A.8. Ejemplo de *Function Block* empleado en el *process model* de un *router*.

Tras llevar a cabo el diseño del FB, se compila el código en busca de errores de programación. OPNET permite otras opciones de programación que ofrecen un enorme número de posibilidades, que no ha sido necesario implementarlas para llevar a cabo las simulaciones de tráfico en la red.

Dominio de Enlace

OPNET dispone de una librería de enlaces donde cada tipo de enlace disponible puede ser configurado. Sin embargo, los enlaces predefinidos no siempre son compatibles con los nodos creados y puede resultar conveniente la creación de un modelo de enlace que pueda interactuar con los nodos.

Los enlaces permiten comunicar información entre nodos mediante la transmisión de paquetes. Un enlace puede estar formado por uno o más canales, cada uno de ellos con su transmisor y receptor correspondientes. En la figura siguiente se muestra la estructura interna de un enlace.

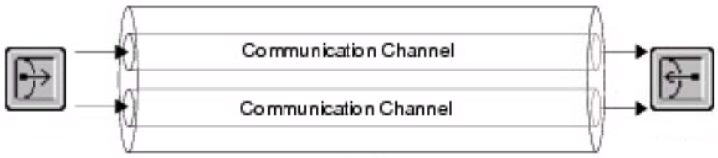

Figura A.9. Enlace, compuesto de uno o más canales independientes.

Para la realización de las pruebas de simulación sobre las herramientas de estimación de capacidad se han creado los modelos de enlace necesarios.

Dominio de Paquete

Los paquetes, al igual que los enlaces en OPNET, se pueden diseñar o emplear los predefinidos por la herramienta. Cada paquete está compuesto de diversos campos que se configuran de manera independiente. Cada paquete consta, principalmente, de nombre, tipo, tamaño y valor por defecto. En la figura A.10 se muestra un tipo de paquete empleado en las simulaciones de tráfico en la red; en él se pueden observar los diferentes campos que lo componen, como pueden ser la ruta a seguir, el tipo, las impresiones temporales, etc.

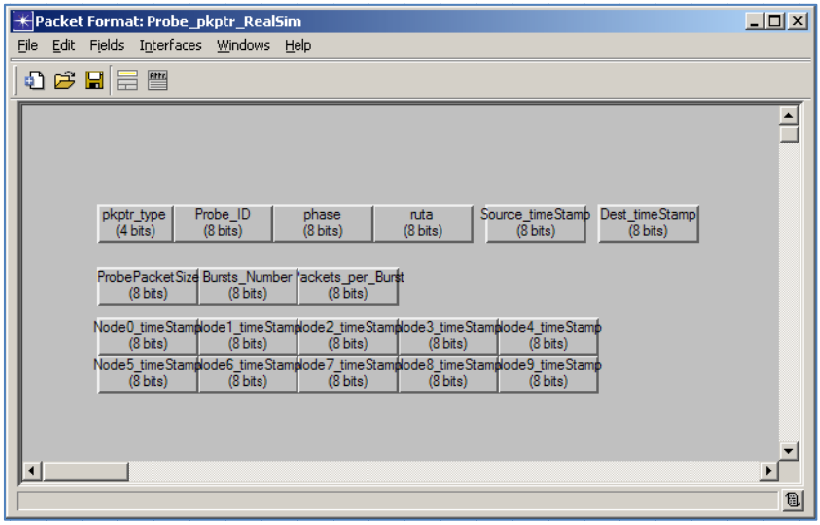

Figura A.10. Formato de paquete.

Dominio de red

Un modelo de red define el objetivo general del sistema a simular. Representa una descripción a alto nivel de los objetos contenidos en el sistema. El nivel de red especifica los objetos en el sistema junto con sus localizaciones físicas, interconexiones

y configuraciones. Un modelo de red puede contener subredes, nodos y enlaces. Una vez que estos elementos han sido implementados, el diseñador puede disponer de ellos e incluirlos en la arquitectura. Algunas de las opciones que el dominio de red permite son la configuración de estadísticas, el perfil de configuración de movilidad o la configuración de simulación de eventos discretos.

Un paso importante a realizar antes de la simulación consiste en la verificación de los enlaces. La biblioteca de modelos de enlaces y nodos de OPNET permite al programa determinar si los enlaces y transceptores están conectados y configurados de forma válida. En la figura siguiente se muestran los dos escenarios configurados en el dominio de red en este proyecto.

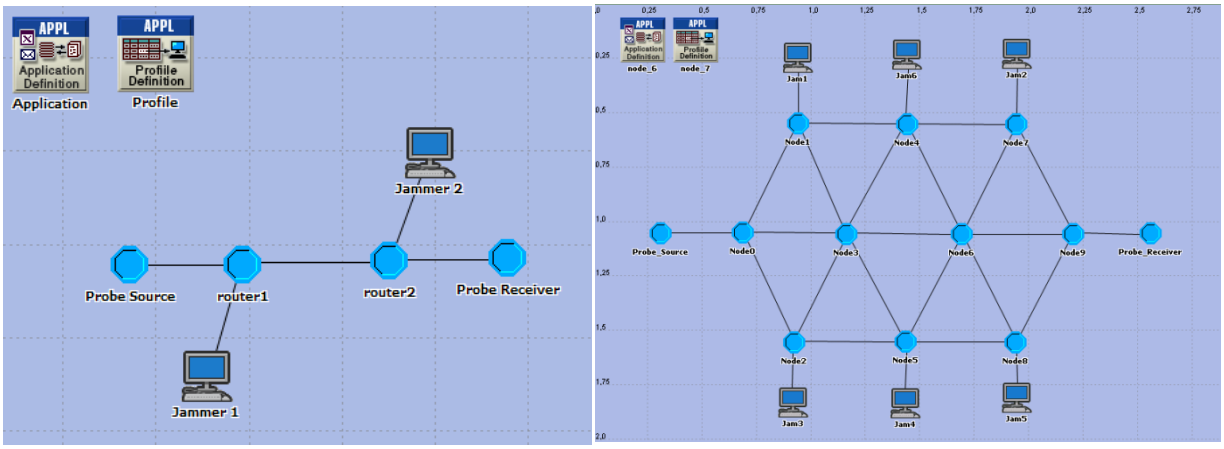

Figura A.11. Escenarios configurados en el nivel de red empleados en la simulación.

OPNET ofrece un interfaz gráfico que permite al diseñador simular arquitecturas de red. El amplio abanico disponible para la configuración de nodos, enlaces y paquetes permite llevar a cabo el diseño de cualquier escenario.

ANEXO III. OTROS RESULTADOS DE SIMULACIÓN

III.1 EQoSIM

III.1.1 Tamaño óptimo de paquete

La técnica empleada en *EQoSIM* para la estimación de la capacidad aplica el método basado en *Packet Pair* más teórico sin modificación. Para su implementación ha sido necesaria la configuración de varios parámetros a partir de medidas empíricas experimentales.

En primer lugar es necesario conocer el tamaño de los paquetes de prueba que serán inyectados en la red, para lo cual se llevan a cabo diversos experimentos sobre los escenarios diseñados. En las dos siguientes figuras se muestra la efectividad de las estimaciones empleando varios tamaños de paquete:

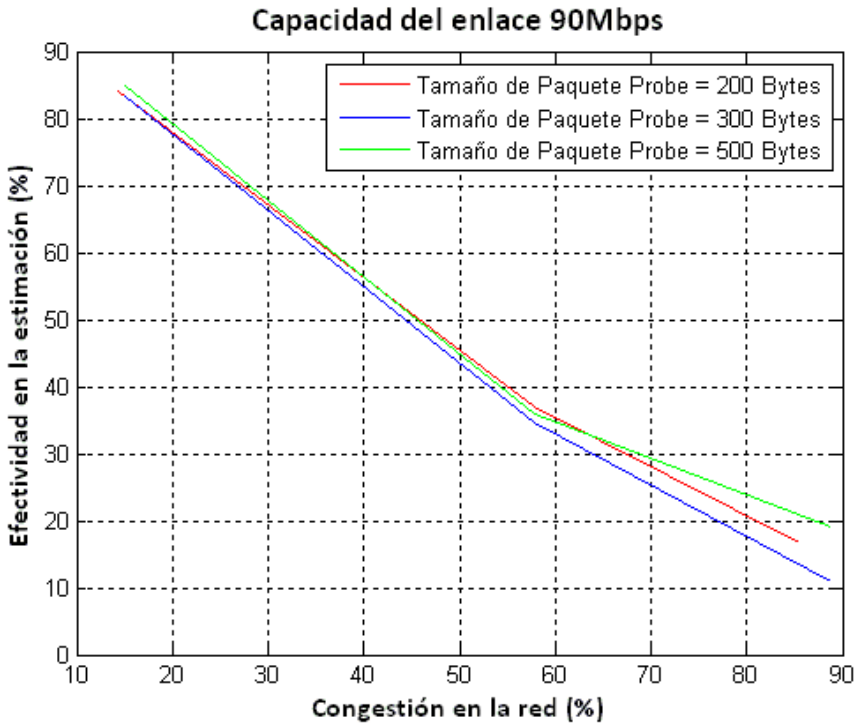

Figura A.12. Efectividad de medidas de capacidad en enlace a 90Mbps

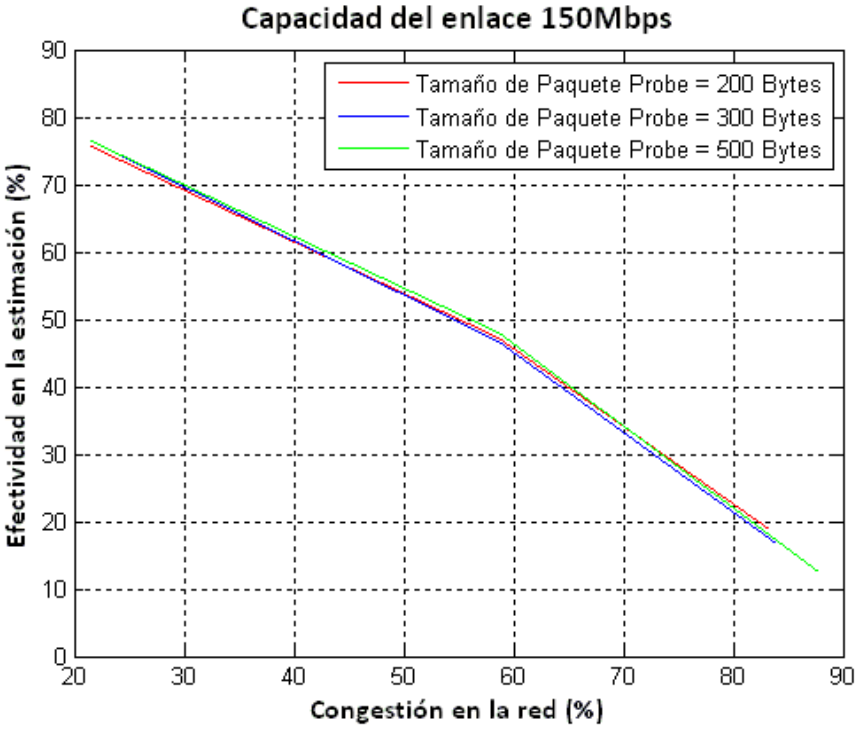

Figura A.13. Efectividad de medidas de capacidad en enlace a 150Mbps

De acuerdo con la filosofía del método se buscará ser lo menos intrusivo posible para no interferir en las comunicaciones existentes. Por ello, y dada la similitud de los resultados vistos en la figura anterior, se escoge un tamaño de paquete adecuado y cuya interferencia sobre otros usuarios sea mínima; la dimensión de *probe packet* óptimo será de 200 Bytes. Otro motivo por el que elegir un tamaño pequeño es, tal y como hemos visto, porque se prefiere que no se encole el *probe* a que caiga mucho la estimación de la capacidad (cuando un paquete *jammer* se introduce entre los dos paquetes del par de prueba).

III.1.2 Longitud óptima de ráfaga

Tras conocer el tamaño de paquete que *EQoSIM* empleará es necesario establecer la longitud de las ráfagas a utilizar, es decir, el número de paquetes consecutivos que se enviarán.

Se han llevado a cabo pruebas de simulación sobre los escenarios diseñados. Se han realizado estimaciones de capacidad mediante de ráfagas de diferente longitud. El resultado es prácticamente el mismo empleando ráfagas que van desde el par hasta la decena de paquetes. La comparación de los resultados obtenidos se puede observar en la siguiente figura.

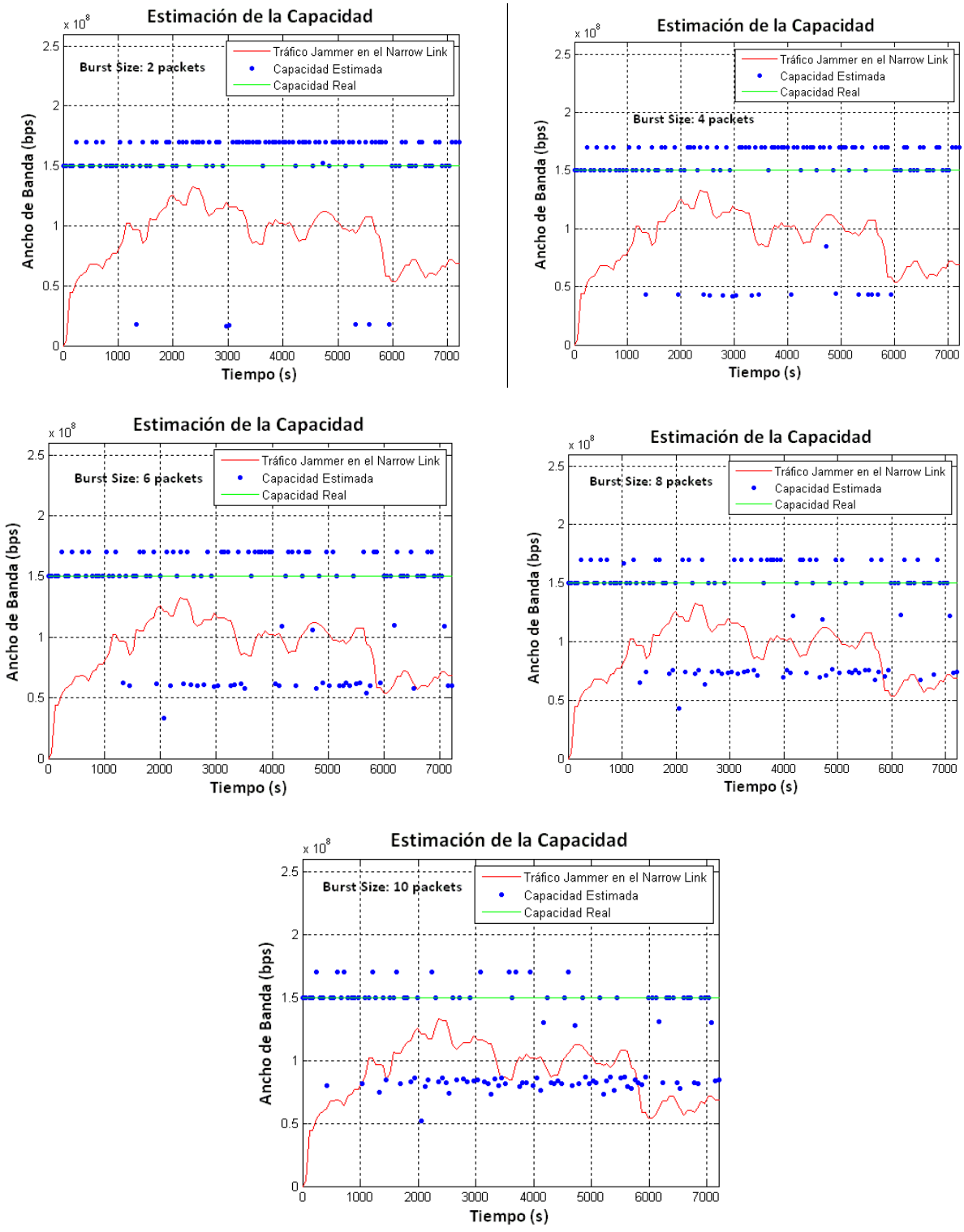

Figura A.14. Estimación de capacidad para 5 longitudes de ráfaga diferentes.

Se observa en la figura como el empleo de diferentes longitudes de ráfaga dan como resultado el mismo número de estimaciones correctas de capacidad (el hecho de que las estimaciones incorrectas varíen entre subestimación y sobreestimación no afecta a la medida). Existe un compromiso entre intrusividad y precisión; a mayor

longitud de ráfaga mayor será la probabilidad de que un paquete interferente llegue a un nodo o *router* entre dos paquetes *probe* de la ráfaga y la estimación resulte incorrecta. Por otro lado, cuanto más larga sea la ráfaga menor será el error en la estimación en caso de que un paquete *jammer* se cuele entre dos paquetes de prueba. El objetivo es lograr el número máximo de estimaciones correctas con la menor intrusividad, por lo que, tras realizar las simulaciones y observar los resultados presentes en la figura A.4, se ha optado por la configuración de ráfaga que realice las estimaciones con menor probabilidad de error; se enviarán ráfagas de dos paquetes.

III.2 Metodología en la Obtención de Resultados

En el Capítulo 4 se describe la metodología para la obtención de una estimación de capacidad (ver figura 13) y en este capítulo se comenta el procedimiento para obtener las figuras de intrusividad, error, retardo y fiabilidad mostradas en el Capítulo 5 a partir de las diferentes estimaciones de capacidad.

Las simulaciones se han llevado a cabo sobre dos escenarios, los cuales se observan en las figuras mostradas a continuación. En el segundo de ellos se han definido tres rutas que seguirán los paquetes de prueba de terminal fuente a terminal destino.

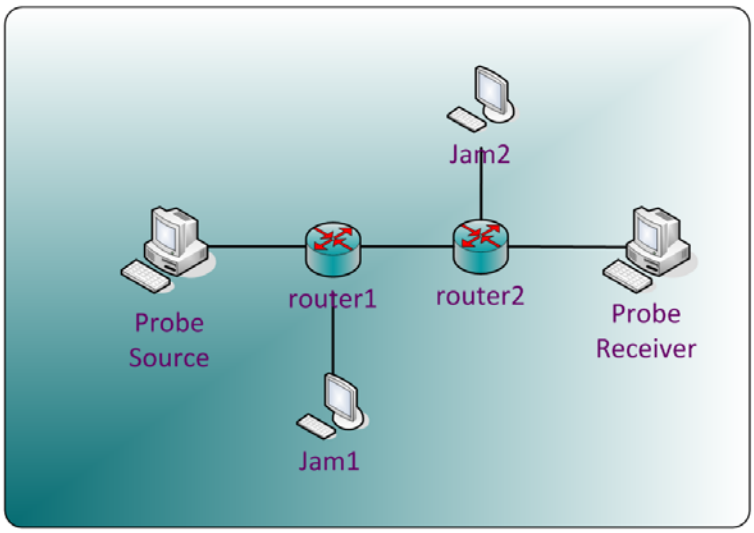

Figura A.15. Escenario 1.

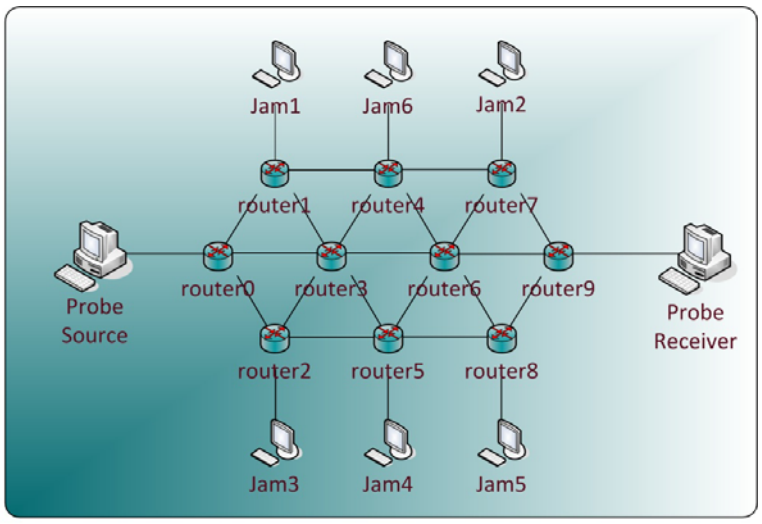

Figura A.16. Escenario 2.

Gracias al diseño modular ha sido posible evaluar cada uno de los cuatro métodos en las mismas condiciones de congestión de la red intercambiando el módulo de la herramienta en los terminales fuente y destino. De cada sesión se obtiene una gráfica como la que se muestra en la figura A.17:

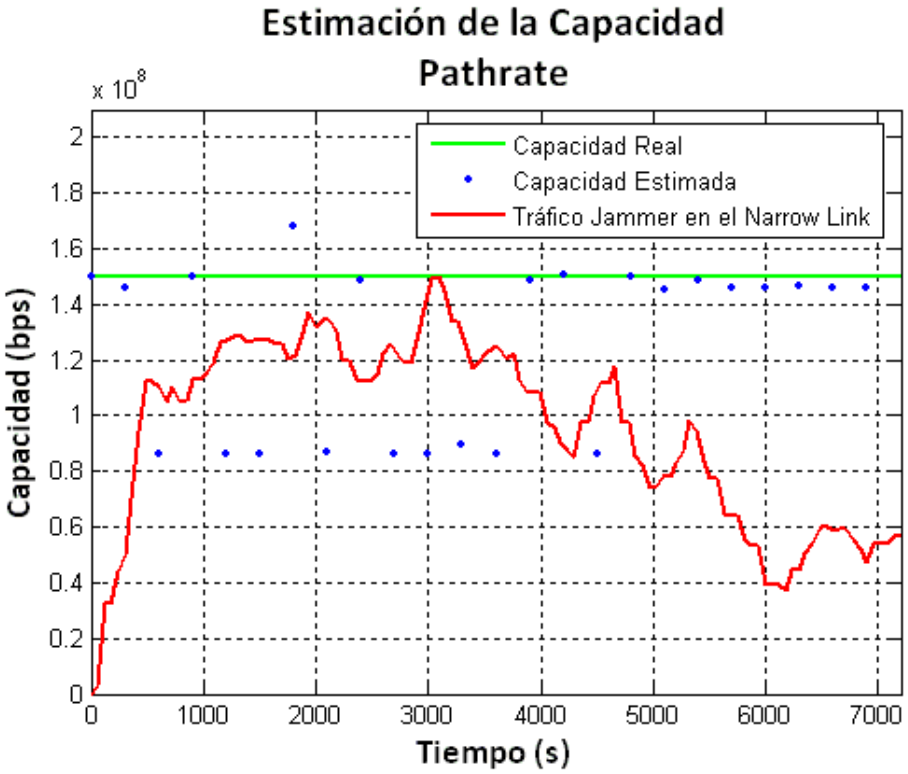

Figura A.17. Resultado de simulación en Media Congestión.

Se pueden observar en la gráfica las estimaciones de capacidad efectuadas cada cinco minutos mediante la herramienta *Pathrate*. Se advierte que no existe tráfico interferente en el comienzo de la simulación y una evolución de éste durante las dos horas de prueba. La capacidad del *narrow link* en este caso es de 150 Mbps. Las estimaciones de capacidad realizadas varían, siendo acertadas durante una gran parte del tiempo y erróneas en los casos en que la congestión es mayor. Las pequeñas oscilaciones experimentadas por las medidas en torno a la capacidad real del enlace están ocasionadas por el modo de funcionamiento de *Pathrate*: la desviación sobre el valor exacto se debe a la anchura de la celda del histograma mediante el cual *Pathrate* agrupa las estimaciones. En la figura A.18 se muestra otra sesión de simulación en la que se prueba el método EQoSIM.

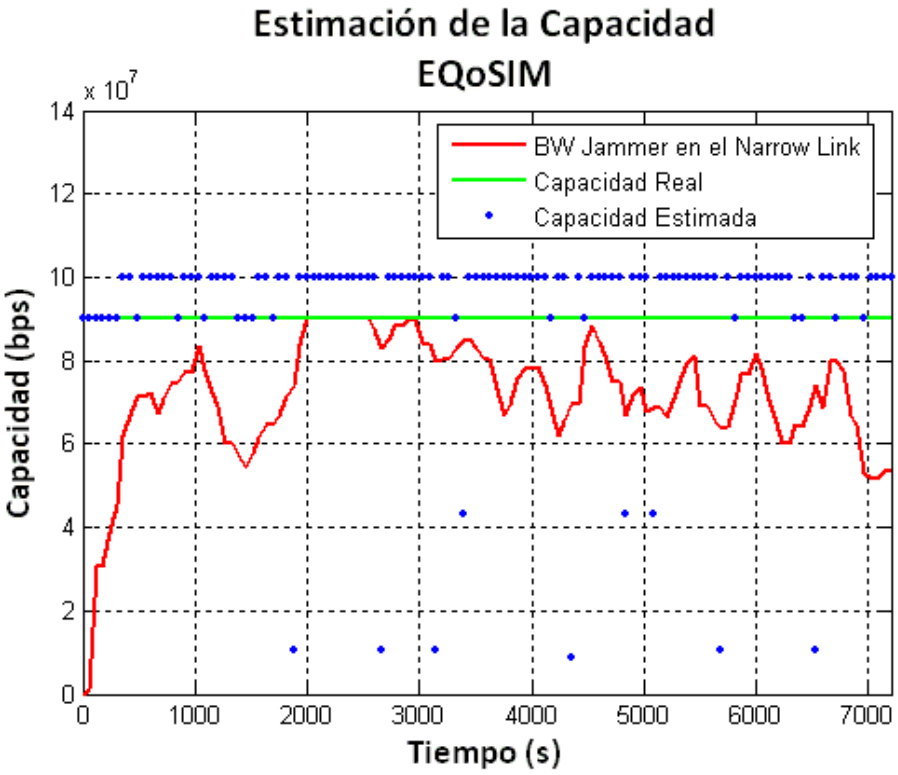

Figura A.18. Resultado de simulación en Alta Congestión.

La figura representa el resultado de una prueba de simulación realizada sobre la herramienta *EQoSIM* en un escenario por cuyo *narrow link* circulan los paquetes a 90Mbps. El grado de congestión es alto, llegando en algunos instantes al 100%. Se observa como las estimaciones son acertadas mientras la interferencia no es demasiado grande, y conforme ésta aumenta aparecen los dos tipos de error, subestimación y sobreestimación, siendo ésta última la que afecta a la herramienta *EQoSIM* en mayor medida, tal y como se aprecia en la figura A.18. Asimismo es posible apreciar cómo, para un grado alto de congestión, la fiabilidad de la herramienta es baja y, sin embargo, el error relativo no se dispara; esto se debe a que se ve influenciado por los enlaces posteriores al *narrow link*, y en este caso estima una capacidad de 100Mbps. Si la capacidad del enlace posterior al *narrow link* en que se produce la sobreestimación fuera mayor de 100 Mbps, el error relativo en esta sesión resultaría mayor.

Del conjunto de simulaciones realizadas sobre cada herramienta se extraen los resultados de estimación de capacidad en función del tráfico interferente. Del análisis de estos resultados se obtienen los valores de retardo, intrusividad, error y fiabilidad que, una vez promediados, proporcionan los resultados comentados anteriormente en el Capítulo 5. La siguiente figura muestra un esquema del proceso:

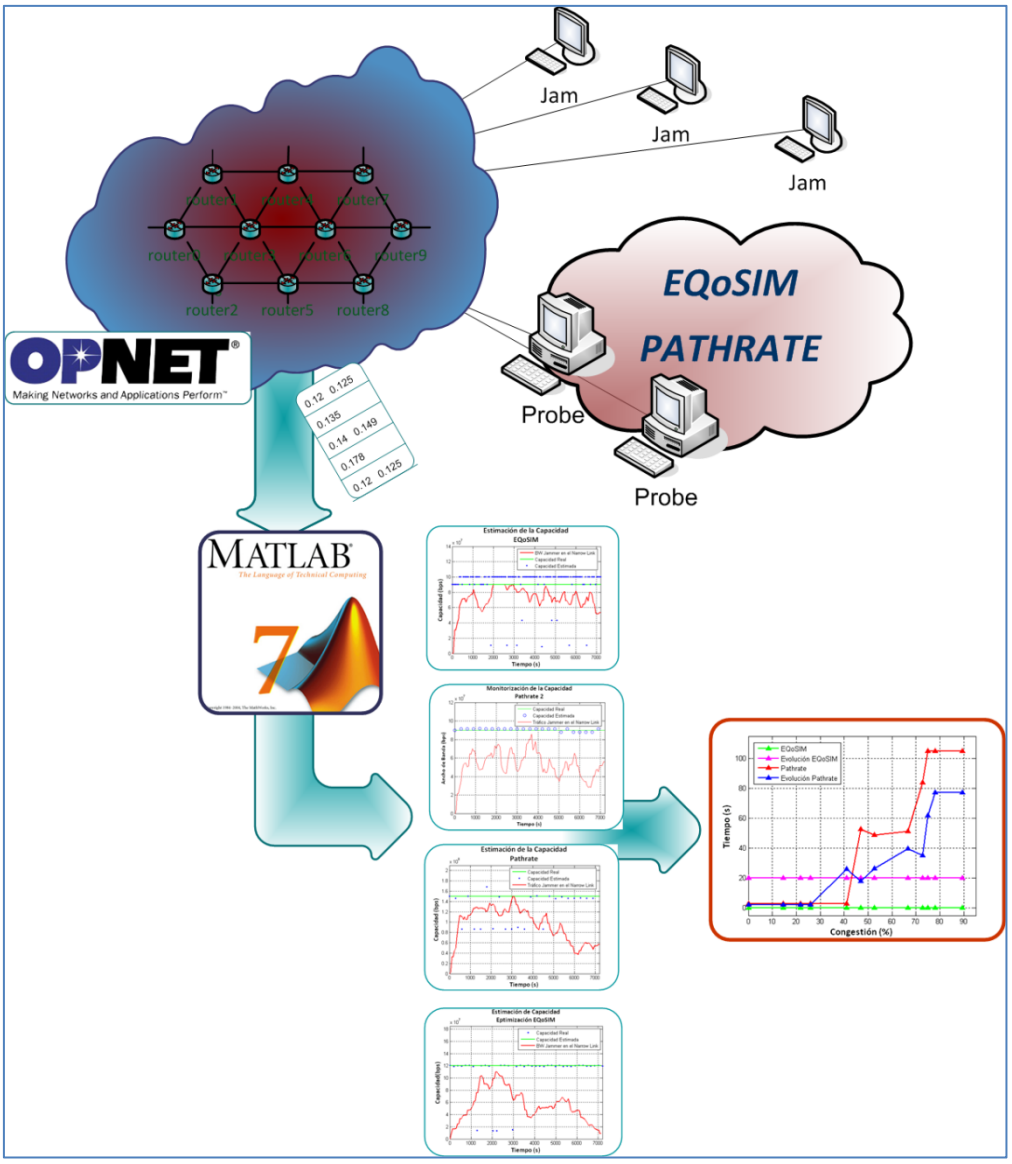

Figura A.19. Proceso de obtención de resultados en función de congestión.