

Protocolos de Enrutamiento Simulador de Tráfico de Redes

Andres Gomez Marquez

¹Universidad Cooperativa de Colombia¹. Bogotá Colombia E-mail:
andresgomez@grupooyg.com.co rangoma@hotmail.com Dpto. Informática; 9 de
Junio - 2008

Área: Personas; Subárea: Educación.

Resumen. Teniendo presente la gran importancia de las redes de datos en la Sociedad de la Información y el Conocimiento, resulta imprescindible facilitar su estudio especialmente en instituciones que no cuentan con la posibilidad de brindar grandes redes reales para que sus estudiantes trabajen y experimenten con ellas; de allí la importancia de las herramientas de simulación de redes, que permiten su estudio bajo diferentes cargas de tráfico, facilitando la experimentación y el estudio personalizado y a distancia de los alumnos. Los protocolos de enrutamiento para la capa de red son usados para resolver peticiones de servicios de envío de paquetes de datos a través de diferentes redes de datos. El punto más importante de este estudio es mostrar el comportamiento de este tipo de tráfico mediante un simulador didáctico y la forma de incorporar automáticamente diferentes configuraciones de redes para el mismo.

Palabras Claves: Protocolos de Enrutamiento – Simulación – Transmisión de Información – RIP

1 Introducción

Teniendo en cuenta las necesidades y los avances producidos en una sociedad sumamente compleja, resulta de gran importancia destacar tanto la transmisión de información, como la necesidad de que ésta llegue a destino en el momento preciso mediante el uso de las redes.

Es a través de la Internet que queda probado, y todos los días se muestra con mejor y mayor detalle, que ésta ha sido y será revolucionaria en las áreas de los servicios financieros, de entretenimiento, salud, educación y gobierno.

El proceso de digitalización de todas las técnicas de comunicación, transmisión (cable, satélite) y recepción, producen nuevas convergencias entre diferentes sectores (cultura, comunicación, lengua, educación, telecomunicaciones, etc.), pero muy especialmente lo que producen es *la transformación de los “espacios de comunicación”*, los límites y las fronteras y, como consecuencia, *la transformación de los espacios de intercambios culturales*. De hecho, todas las sociedades, por definición, han sido y serán “sociedades de la comunicación” [2].

Los principales cambios estructurales de la sociedad se producen ahora entorno del tratamiento y de la transmisión de la información.

La capa de red, dentro de una arquitectura de red de datos, es la que se encarga de llevar los paquetes de datos desde el origen (estación transmisora) hasta el destino (estación receptora). Llegar a destino, en tiempo y forma, puede requerir que el algoritmo de ruteo, que es el encargado de escoger las rutas y las estructuras de datos, cumpla con ciertas propiedades que aseguren la eficiencia de su trabajo.

Estas propiedades son: *corrección, estabilidad, robustez, equitatividad, sencillez y optimalidad* [1].

La corrección y la sencillez casi no requieren comentarios; no así la necesidad de robustez, la cual se refiere a que el algoritmo debe ser diseñado para que funcione dentro de la red por años, sin fallas generales. El algoritmo deberá estar preparado para manejar cambios de topología y tráfico sin requerir el aborto de las actividades o el rearranque de la red.

La equitatividad y la optimalidad resultan con frecuencia contradictorias, ya que muchas veces se requiere una concesión entre la eficacia global (optimización) y la equitatividad; es decir, antes de intentar encontrar un justo medio entre estas dos, se debe decidir qué es lo que se busca optimizar.

Minimizar el retardo de los paquetes (disminuyendo escalas y ancho de banda) y maximizar el rendimiento total de la red sería la combinación más apropiada para un algoritmo de ruteo.

A los efectos de facilitar el estudio de aspectos relacionados con el tráfico en las redes se utilizan simuladores didácticos, que permiten analizar el comportamiento de los algoritmos de ruteo para distintas configuraciones de redes, bajo diferentes tipos de cargas de tráfico.

2 La Capa de Red en Internet

En la capa de red, la Internet puede verse como un conjunto de subredes, o *sistemas autónomos (AS)* interconectados. No hay una estructura real, pero existen varios *backbone* principales. Estos se construyen a partir de líneas de alto ancho de banda y enrutadores rápidos. Conectadas a los *backbone* hay redes regionales (de nivel medio), y conectadas a estas redes están las LAN de muchas universidades, compañías y proveedores de servicios de Internet.

El “pegamento” que mantiene unida la Internet es el protocolo de la capa de red, *IP* (*Internet Protocol*, Protocolo de Internet). Este protocolo se diseñó desde sus principios con la interconexión de redes en mente. Debe proporcionar el “mejor esfuerzo” para el transporte del datagrama del origen hacia el destino, sin importar si éstos están en la misma red o si hay otras redes entre ellos [3].

Protocolo de información de ruteo: RIP

Como se ha dicho antes, la Internet se compone de una gran cantidad de sistemas autónomos (AS). Cada AS es operado por una organización diferente y puede usar internamente su propio algoritmo de enrutamiento. Por ejemplo, las redes internas de las compañías X, Y, y Z generalmente se verían como tres AS si las tres estuvieran en Internet. Las tres pueden usar algoritmos de enrutamiento diferentes internamente. No obstante, la existencia de estándares aún para enrutadores internos, simplifica la implementación en las líneas divisorias entre los AS y permite la reutilización de código. El algoritmo de enrutamiento interno de un AS se llama *protocolo de pasarela interior*; al algoritmo de enrutamiento entre varias AS se le llama *protocolo de pasarela exterior*.

El protocolo RIP [7] [8], al igual que sus antecesores propietarios es un protocolo de ruteo que fue diseñado para funcionar como protocolo “vector distancia”. RIP fue diseñado para funcionar en redes pequeñas de pasarela interior. RIP está basado en la versión 4.3 de la distribución de UNIX de Berkeley.

En cuanto al protocolo tenemos que tener en cuenta las siguientes tres limitaciones:

- 1 El protocolo no permite más de quince saltos, es decir, los dos enrutadores más alejados de la red no pueden distar más de 15 saltos, si esto ocurriera no sería posible utilizar RIP en esta red.
- 2 Problema del “conteo a infinito”. Este problema puede surgir en situaciones atípicas en las cuales se puedan producir bucles, ya que estos bucles pueden producir retardos e incluso congestión en redes en las cuales el ancho de banda sea limitado.
- 3 El protocolo utiliza métricas fijas para comparar rutas alternativas, lo cual implica que este protocolo no es adecuado para escoger rutas que dependan de parámetros a tiempo reales como por ejemplo retardos o carga del enlace.

RIP es un protocolo que genera muchísimo tráfico al enviar toda la tabla de ruteo en cada actualización, con la carga de tráfico que ello conlleva.

Tabla de ruteo del RIP

La base de datos de ruteo de cada uno de los hosts de la red que utilizan el protocolo de ruteo RIP tiene los siguientes campos:

1. Dirección de destino.

La dirección de destino en la tabla de ruteo de RIP será la red de destino, es decir, la red final a la que se desea acceder; esta red en la versión 1 del protocolo RIP tendrá que ser obligatoriamente classfull, es decir tendrá que tener en cuenta la clase, es

decir, no se permite el subneting en RIP versión 1, por ejemplo si la red de destino es la 192.168.4.0, se sabe que al ser RIP classfull la red de destino tiene 256 direcciones, de las cuales 254 son útiles, una vez descontada la dirección de red y la dirección de broadcast, ya que la red 192.168.4.0 es de clase C, es decir que los 24 primeros bits de la dirección IP identifican la red y los 8 últimos identifican los hosts de dentro de la red.

2. *Siguiente salto.*

El siguiente salto se define como el siguiente enrutador por el que el paquete va a pasar para llegar a su destino, este siguiente salto será necesariamente un enrutador vecino del enrutador origen.

3. *Interfaz de salida del enrutador.*

Se entiende por interfaz de salida del enrutador a la interfaz a la cual está conectado su siguiente salto.

4. *Métrica.*

La métrica utilizada por RIP consiste en el conteo de saltos; se considera cada salto como una única unidad, independientemente de otros factores como tipo de interfaz o congestión de la línea. La métrica total consiste en el total de saltos desde el enrutador origen hasta el enrutador destino, con la limitación que 16 saltos se considera destino inaccesible, lo cual limita el tamaño máximo de la red.

5. *Temporizador.*

El temporizador indica el tiempo transcurrido desde que se ha recibido la última actualización de cierta ruta. RIP utiliza varios tiempos importantes, el tiempo de actualización que se establece en 30 segundos, el tiempo de desactivación que se establece en 180 segundos y el tiempo de borrado se establece en 300 segundos.

El *tiempo de actualización* es considerado el tiempo máximo a transcurrir entre el envío de los mensajes de actualización de los vecinos.

El *tiempo de desactivación* se considera como el tiempo máximo que puede esperar un enrutador sin recibir actualizaciones de un vecino, una vez pasado este tiempo, el vecino que no ha enviado la actualización se considera que se ha caído, con lo cual el enrutador no está activo en la red, por lo cual se establece la métrica a valor 16, es decir destino inalcanzable.

El *tiempo de borrado* implica que una vez transcurrido ese tiempo todas las rutas de ese enrutador supuestamente caído son eliminadas de la tabla de enrutamiento.

Para obtener esta tabla, el protocolo de enrutamiento RIP utiliza el siguiente procedimiento para mantener actualizada la tabla de enrutamiento de cada uno de los nodos o enrutadores de la red:

1. Mantener una tabla con una entrada por cada posible destino en la red. La entrada debe contener la distancia D al destino, y el siguiente salto S del enrutador a esa red. Conceptualmente también debería de existir una entrada para el enrutador mismo con métrica 0, pero esta entrada no existirá.
2. Periódicamente se enviará una actualización de la tabla a cada uno de los vecinos del enrutador mediante la dirección de broadcast. Esta actualización contendrá toda la tabla de enrutamiento.

3. Cuando llegue una actualización desde un vecino S , se añadirá el coste asociado a la red de S , y el resultado será la distancia D' . Se comparará la distancia D' y si es menor que el valor actual de D a esa red entonces se sustituirá D por D' .

Protocolo de enrutamiento de pasarela interior: OSPF

El protocolo de pasarela interior original de Internet fue un protocolo de vector de distancia (RIP) basado en el algoritmo Bellman-Ford. Este protocolo funcionó bien en pequeños sistemas, pero menos bien a medida que los AS se volvieron más grandes. En 1988, la *Internet Engineering Task Force* (Grupo de Trabajo de Ingeniería de Internet) comenzó a trabajar en su sucesor. Ese sucesor llamado OSPF (*Open Shortest Path First*, abrir primero la trayectoria más corta), se convirtió en estándar en 1990 [9] [10].

El OSPF funciona haciendo una abstracción del conjunto de redes, enrutadores y líneas en un grafo dirigido en el que a cada arco se le asigna un costo (distancia, retardo, etc.). Entonces se calcula la trayectoria más corta con base en los pesos de los arcos.

Lo fundamental que hace el OSPF es representar la red como un grafo y luego calcular la trayectoria más corta de un enrutador a todos los demás.

El OSPF funciona intercambiando información entre enrutadores *adyacentes*, que no es lo mismo que entre enrutadores vecinos. En particular es ineficiente hacer que todos los enrutadores de una LAN hablen con todos los enrutadores de otra LAN.

Para evitar esta situación, se elige un enrutador como *enrutador designado*, el cual se dice que es adyacente a todos los demás enrutadores, e intercambia información con ellos. Los enrutadores que no son vecinos no se intercambian información entre ellos, se mantienen actualizados designando un enrutador de respaldo para facilitar la transición en el caso que el enrutador designado primario se caiga.

3 Flan

Se define simulación como la técnica de reproducir la esencia de un fenómeno sin reproducir el fenómeno en sí, con la ventaja adicional de poder hacerlo en una escala de tiempo muy pequeña o comprimida, si nos valemos de una computadora para realizar su procesamiento [4].

La simulación se ha utilizado para diseñar y para probar las nuevas ideas de los desarrolladores por años. Sin embargo, el proceso de la simulación no es perfecto y como con los simuladores de cualquier naturaleza, nada es tan real como las cosas verdaderas. Las ventajas provienen de poder crear, diseñar, experimentar y destruir en un ambiente que reconstruya exactamente los eventos que son simulados en una fracción de su verdadero costo.

Flan es una herramienta de simulación basada en Java que permite el diseño, la construcción, y la prueba de una red de ordenadores en un ambiente simulado. Hace un acercamiento con un alto nivel de abstracción de una red haciendo la generalización que una red está compuesta de los bloques simples conocidos como enlaces (links) y nodos. Se ha utilizado este simulador por sus características

multiplataforma, que permiten su utilización en distintos sistemas operativos que soporten máquina virtual Java [5], como así también por la sencillez de su utilización, lo que facilita su empleo en el proceso de enseñanza – aprendizaje de la problemática de los protocolos de ruteo de la capa de red.

Flan está pensado para la prueba de protocolos y redes pequeñas (menos de 100 nodos). Aunque el usuario puede tener tantos nodos como desee, el funcionamiento se verá afectado mientras se agreguen más y más nodos.

Para todo enlace de *Flan*, el ancho de banda, el retraso, y los nodos extremos son especificados en su creación, configurando la conexión requerida. Para el usuario, esto se comporta como distintos tipos de conexiones, pero para el simulador, un enlace es un enlace y sus variables son las únicas consideraciones tenidas en cuenta.

Un nodo toma las mismas generalidades. Ya sea una pasarela, un host, o un hub, cada nodo es tratado como el mismo tipo de objeto, y esto permite mayor facilidad a la hora de personalizar las conexiones.

Las diferencias entre los nodos son determinadas por factores tales como el número de conexiones al nodo y a los manejadores de protocolos asignados.

Los eventos de usuario (user-events) permiten que éste controle la simulación y los eventos que ocurren en la red simulada. Estos eventos pueden ser escritos por el usuario utilizando Java y puestos en ejecución en una red diseñada y construida en *Flan*.

Los requerimientos previos al uso y desarrollo de este programa son un kit de desarrollo de Java tal como el kit estándar de desarrollo de la edición de la plataforma de Java 2 (J2SE). Esto tiene la máquina virtual necesaria de Java (JVM), interfaz de programación de base (APIs), y el compilador que es necesario para compilar y desarrollar en *Flan*.

Una vez que el kit de desarrollo requerido está instalado, la máquina virtual de Java permite que *Flan* funcione en cualquier sistema que contenga el JVM.

El área de trabajo de este simulador consta de tres módulos bien identificados: la *ventana principal* (figura N° 1), la *consola* (figura N° 2) y una interfaz de salida gráfica llamada *YO!* (figura N° 3).

Ventana Principal

La ventana principal contiene todos los componentes necesarios para crear, destruir, y manipular una red. Esto incluye la *barra de menú*, la *barra de herramientas*, y la *hoja de dibujo*.

La barra de menú contiene herramientas útiles que le permiten al usuario limpiar la hoja de dibujo y comenzar otra vez, guardar una red existente o abrir una grabada previamente. También se dispone en la barra de menú de una opción para cargar un escenario previamente hecho. Un escenario es un archivo que contiene los eventos que pueden ocurrir en la red, que el usuario ha programado previamente.

La hoja de dibujo es donde se dibuja la red para ser simulada; los nodos y los links se disponen para proporcionar una representación gráfica de la red prevista por los usuarios.

La mayoría de las herramientas y elementos usados en el proceso de la simulación están situados en la barra de herramientas, que en este simulador está compuesta por botones donde la función de cada uno de ellos está indicada por una imagen.

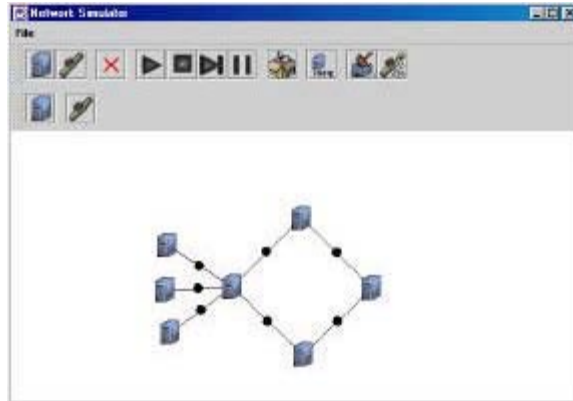


Figura N° 1: Hoja de Dibujo.

Consola

La consola muestra al usuario la información de la red y proporciona pistas en cuanto a qué está sucediendo en tiempo del simulador. Mediante este módulo el usuario tiene la capacidad de ver no solamente los eventos ocurridos en la hoja de dibujo, sino también analizar y seguir las acciones que esos eventos producen y mucha otra información útil para él.

```
Console
Info: CreatePoissonEventHandler: HANDLER
Info: At time 0.0
Info: 255.168.116.1/255.255.255.0 handling a packet from 255.168.116.1 to 255.168.116.2
Info: Unknown
Info: Packet: Unknown
-- STEP 1.0
-- STEP 2.0
Info: CreatePoissonEventHandler: HANDLER
Info: At time 2.0
Info: 255.168.116.3/255.255.255.0 handling a packet from 255.168.116.3 to 255.168.116.1
Info: Unknown
Info: Packet: Unknown
Info: CreatePoissonEventHandler: HANDLER
Info: At time 2.0
Info: 255.168.116.1/255.255.255.0 handling a packet from 255.168.116.1 to 255.168.116.3
Info: Unknown
Info: Packet: Unknown
-- STEP 3.0
Info: CreatePoissonEventHandler: HANDLER
Info: At time 3.450004
Info: 255.168.116.1/255.255.255.0 handling a packet from 255.168.116.1 to 255.168.116.2
Info: Unknown
Info: Packet: Unknown
Info: CreatePoissonEventHandler: HANDLER
Info: At time 3.6773052
```

Figura N° 2: Consola con Datos.

Interfaz de Salida Gráfica

La interfaz de salida YO! permite al usuario analizar lo sucedido durante la simulación en forma gráfica, mediante un eje de coordenadas, donde las ordenadas representan el número de paquetes y las abscisas el tiempo.

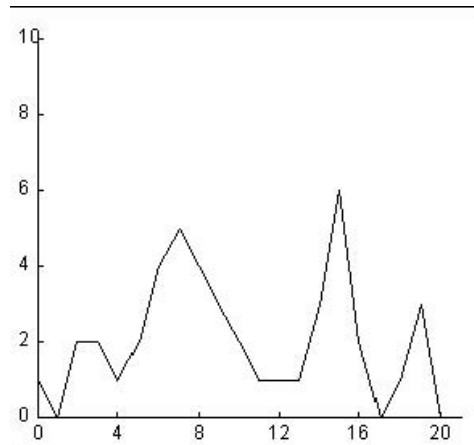


Figura N° 3: Interfaz Gráfica YO!.

Configuración de Paquetes

Eventos

Los paquetes de eventos contienen eventos que el usuario puede invocar en la red simulada. Estos eventos incluyen:

1. Creación de paquetes que se enviarán sobre la red.
2. Ejecución del RIP para actualizar una simple tabla de encaminamiento de un nodo en la red o de todos los que conformen la red.

Al ejecutar la preparación de estos eventos, se le pedirá al usuario proporcionar más información para introducirlos correctamente en la simulación. Por ejemplo, esta información puede ser el tiempo en el que el evento debe ocurrir, a qué nodo específico pertenece el evento, o qué protocolo debe ser utilizado.

Escenario

El escenario proporciona la Interfaz Gráfica de Usuario (GUI) asociada al simulador Flan. Su propósito es proveer al usuario del feedback, del espacio de dibujo y de planeamiento, los mensajes, la información, la entrada, y otros aspectos asociados a proporcionar facilidad de empleo y entendimiento, y al mismo tiempo, hacer que Flan sea satisfactorio para el usuario.

El escenario contiene todo lo que el usuario ve. Si es un ícono del nodo en la hoja de dibujo o una caja de la información que demuestra la tabla de encaminamiento actual, el escenario provee al usuario los objetos gráficos que ayudan en el proceso del diseño y del desarrollo. Además, como el lenguaje Java es multi-plataforma, la

vista y la sensación del escenario varía entre los distintos sistemas pero las funcionalidades son las mismas.

Manejadores

Los manejadores son protocolos específicos y ayudan a determinar cómo es recibida la información, si por parte del usuario o internamente, cómo procesar y además cómo proceder al dirigir la simulación.

Los manejadores podrían incluir ProtocolIP por ejemplo, que conduce la simulación hacia el mundo del IP. Esto incluiría tomar datos abstractos tales como entradas y direcciones de la tabla de encaminamiento, y el proceso de ellas según el estándar del IP. También se incluye en este paquete DefaultHandlers, donde cada nodo se puede asignar a uno o muchos, y funciona para informar a nodos individuales cómo responder a cada protocolo. Si un usuario desearía implementar el encaminamiento basado en IP en una red, estaría obligado a cargar en todos los nodos de esa red el ProtocolIPDefaultHandler, que emplearía ProtocolIP para tomar los datos abstractos y para procesarlos según estándares del IP.

Los manejadores también incluyen paquetes de datos para distintos tipos de datos.

Configuración de la Interfaz entre el Nodo y el Enlace

Toda la configuración de los nodos con respecto al protocolo IP ocurre en la caja de diálogo que se presenta en la figura N° 6. Se utiliza para fijar las direcciones IP de cada interfaz y para llenar la tabla de encaminamiento de cada nodo.



Figura N° 6: Caja de Diálogo del Protocolo IP.

4 GFlan

Con este software desarrollado para complementar al simulador Flan se puede diseñar y construir una red con distintos protocolos en un ambiente muy amigable para el usuario.

Las principales características incluyen:

- 1 Multi-Plataforma: esta aplicación, al estar programada en Java tiene la ventaja de poder funcionar en cualquier máquina con soporte de Java.
- 2 Configurable: deberá existir una manera de alterar los parámetros de la red, tanto de los nodos como de los enlaces.
- 3 Compatible: exporta las redes en formato XML.

Este programa ofrece al usuario la capacidad de “instalar” dentro de cada uno de los enlaces y nodos, variables que agregan facilidades y flexibilidad. Por ejemplo, variables tales como el ancho de banda y el retraso en un enlace pueden ser manejadas y cambiadas en esta aplicación. Y en cuanto a los nodos, la posibilidad de tratarlos dentro de la red como una terminal o como un enrutador.

Esto permite un alto nivel de arreglos para requisitos particulares.

Configuración de un Nodo

Permite al usuario asignar un nombre específico a cada nodo. Cada uno tendrá nombres diferentes, ya que la aplicación les asigna automáticamente un número distinto. La razón de esto es poder identificar un nodo en particular por su nombre. Esto ayuda a no crear confusiones cuando una red contiene muchos protocolos y direcciones. El usuario también deberá elegir la función que el nodo tendrá dentro de la red (terminal o enrutador). Existe la opción de elegir el protocolo con el que se trabajará a partir de una lista de protocolos disponibles que GFlan proporciona.

En la tabla de *direcciones de destino* el usuario puede agregar las direcciones IP destino, junto con cada una de máscaras. A medida que se van creando los nodos, toda la información referente a ellos se irá mostrando en la consola de la ventana principal, como se ve en la figura N° 7.

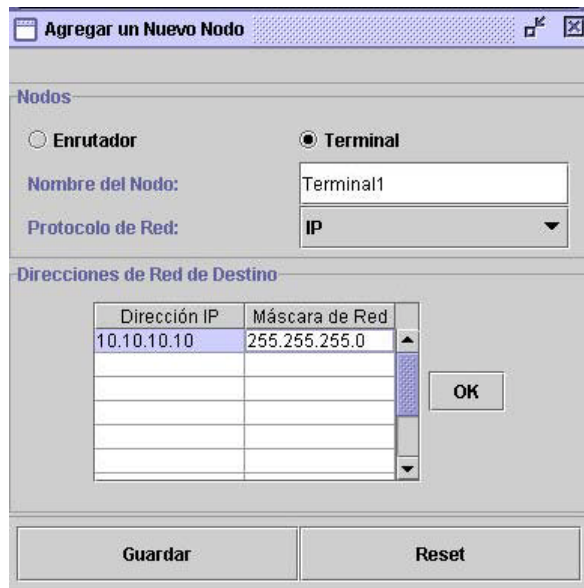


Figura N° 7: Ventana de Configuración de un Nodo.

Configuración de un Enlace

Esta ventana contiene la información que pertenece a los diversos enlaces de una red. La figura N° 8 muestra la ventana de configuración de un enlace que permite al usuario modificarlos para requisitos particulares de una red definiendo el nombre, el ancho de banda y el retraso asociado con él.

El ancho de banda es la cantidad de datos que se puedan pasar sobre el enlace por segundo.

El retraso es el tiempo que toma la información para llegar desde un destino a otro.

Figura N° 8: Ventana de Configuración de un Enlace.

Estructura de Datos del Archivo XML

La gran capacidad para representar datos complejos, como así también la posibilidad de separar la estructura de los documentos del contenido y de la presentación por parte del lenguaje XML, lo hace ideal para estructurar los datos de la red y facilitar el intercambio de documentos entre aplicaciones [6].

La estructura de datos del documento XML donde se almacenan los datos de la red a simular, aparece como una jerarquía estrictamente anidada de elementos. Los elementos tienen atributos y pueden contener texto u otros elementos como hijos.

Sólo se puede almacenar una red por documento XML. El elemento red será la raíz del documento XML y se denotará con el tag:

```
<Network>
</Network>
```

En un nivel más interior tenemos dos nuevos elementos la interfaz del nodo y la interfaz del enlace.

```
<Network>
<InterfaceNode>
</InterfaceNode>
<InterfaceLink>
</InterfaceLink>
```

</Network> En un diagrama jerárquico queda representado como muestra la figura N° 9.

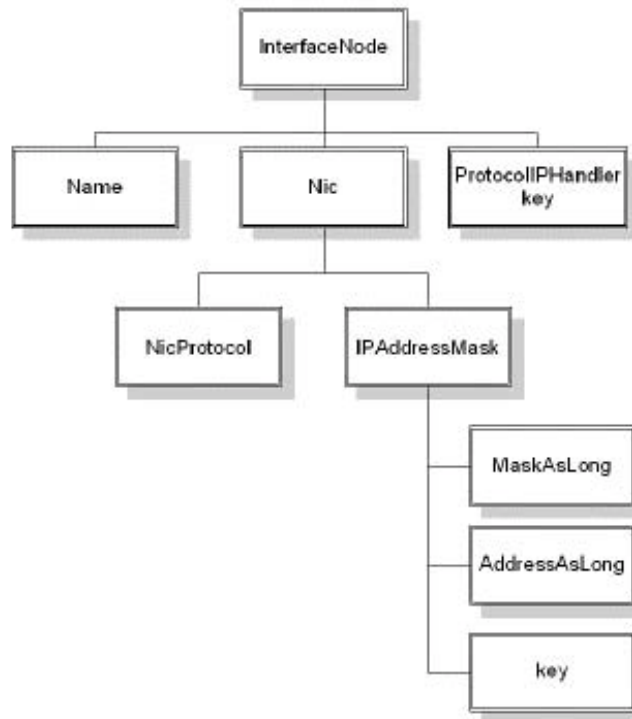


Figura N° 9: Diagrama de Jerarquía del Elemento InterfaceNode.

Interfaz del Nodo

Este elemento está jerarquizado en cuatro niveles y contiene información acerca de la interfaz gráfica del nodo, que lo definirá finalmente en el simulador Flan. La posición en la pantalla está dada por los siguientes elementos:

<xPlacement> 125 </xPlacement>

el cual da la posición vertical del nodo y:

<yPlacement> 150 </yPlacement>

que define la posición horizontal del mismo. La imagen que representará al nodo ya sea terminal o enrutador está dada por la marca:

```
<Iconname>iconos/nodo.gif</Iconname>
```

El elemento que define el nombre del nodo dentro de la red, es almacenado de la siguiente forma:

```
<Name>Enrutador1</Name>
```

Dentro de la estructura de la interfaz del nodo figura como un hijo de ésta el elemento nodo que es el contenedor de las características propias del nodo, como ser los NICs, que son Interfaces específicas para cada nodo. Si un nodo está conectado con otros dos nodos, tendrá dos eventos, a cada uno le corresponderá un NIC.

Básicamente, un NIC es un evento que es identificado específicamente por el nodo y al que se puede asignar una dirección única. Por lo tanto, esto crea un NIC del objeto, que contiene la información sobre los eventos y sus direcciones.

Dentro de este atributo figuran la marca que identifica el protocolo del NIC, el elemento contenedor de la máscara de red y la dirección IP del nodo destino, a su vez este elemento incluye una clave con la que es posible enlazar, con la interfaz del enlace.

Interfaz del Enlace

Este elemento está jerarquizado en diez niveles y contiene información acerca del enlace y de la interfaz gráfica del mismo, que lo representará en el simulador Flan.

El tag Name indica el nombre del enlace. Las marcas x1Placement y y1Placement, indican la posición del extremo origen del enlace, y las marcas x2Placement y y2Placement determinan la posición del extremo destino del enlace.

El costo está definido por la etiqueta Cost, el tag Iconname indica la ruta donde está almacenada la imagen que representa al enlace.

El retraso y el ancho de banda se indican con las etiquetas Delay y Bandwith, mientras que la etiqueta Link permite relacionar el enlace con la dirección IP de destino.

```
<InterfaceLink>
  <Name>Enlace1</Name>
  <x2Placement>201</x2Placement>
  <x1Placement>78</x1Placement>
  <Cost>0.0</Cost>
  <Iconname> iconos/link.gif</Iconname>
  <Delay>1.0E-5</Delay>
  <Bandwidth>193000.0</Bandwidth>
  <Link>
    <LinkId>617280</LinkId>
  </Link>
```

```
<y2Placement>207</y2Placement>  
<y1Placement>104</y1Placement>  
</InterfaceLink>
```

Por último la estructura jerárquica de la interfaz del enlace se muestra en la figura N° 10.

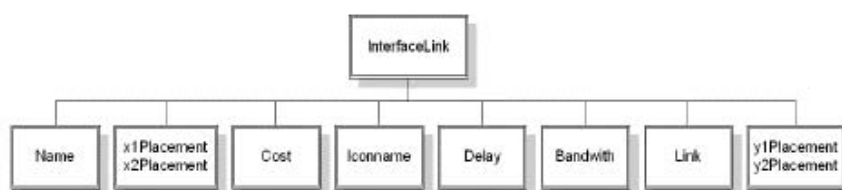


Figura N° 10: Diagrama de Jerarquía del Elemento InterfaceLink.

5 Conclusiones

A través de la realización de este trabajo se pudo probar la validez del simulador Flan al lograr reproducir la esencia del fenómeno de la transmisión de información a través de redes de datos logrando una abstracción con un equilibrio entre la simplicidad y la exactitud, facilitando el estudio de la problemática de los protocolos de ruteo en la capa de red.

A partir de esto se desarrolló una aplicación (GFlan) que permite la creación de diferentes modelos de redes de una forma más amigable e intuitiva para el usuario que la proporcionada por el propio simulador en un formato ampliamente utilizado en la actualidad como es el XML y soportado en un lenguaje multiplataforma muy usado en los ambientes académicos y comerciales como lo es Java.

Se ha podido comprobar que la utilización de la ampliación Gflan como complemento del simulador Flan resulta especialmente provechosa como consecuencia de su amigabilidad al momento de confeccionar los archivos de formato XML utilizados por el simulador Flan, liberando al usuario final, generalmente un docente ó un estudiante de la problemática del ruteo en la capa de red, de la necesidad de conocer la estructura del formato XML para preparar sus archivos de simulación, permitiéndoles dedicarse a la realización de las simulaciones de tráfico y al estudio de los resultados de las mismas.

Asimismo y luego de un gran número de simulaciones efectuadas con diferentes cargas de tráfico y de configuraciones de red, se pudo concluir que la cantidad de paquetes de información de ruteo que se envían por la red no depende solamente de la cantidad de nodos por la que está compuesta, sino que se ve muy influenciada por la complejidad y la manera en que están conectados y agrupados los nodos en subredes.

6 Referencias

- [1] A. S. Tanenbaum. *Redes de Computadoras*. Prentice Hall Hispanoamericana S. A., México-México, 1997.
- [2] L. Joyanes Aguilar. *La Carrera Mundial por el Conocimiento. Una Visión desde la Nueva Economía*. Separata de Corintios XIII, núm. 96, Madrid-España, 2000.
- [3] W. Stallings. *Data and Computer Communications* -Fifth Edition. Prentice Hall, NJ-USA, 1997.
- [4] G. Pace. *Modelos y Simulación*. UNNE, Corrientes-Argentina, 1998.
- [5] E. Castillo; A. Cobo; P. Gómez; C. Solares. *JAVA -Un Lenguaje de Programación Multiplataforma para Internet*. Paraninfo, España, 1997.
- [6] A. Kal; D. Ayers; M. Birbeck; J. Cousins; D. Dodds; J. Lubell; M. Nic; D. Rivers-Moore; A. Watt; R. Worden; A. Wrightson. *Professional XML Metadata*. Wrox Press Ltd., Birmingham-UK, 2001.
- [7] C. Hedrick. *RFC 1058*. Rutgers University, NJ-USA, 1988.
- [8] G. Malkin. *RFC 2453*. Bay Networks, USA, 1998.
- [9] J. Moy. *OSPF Version 2, RFC 1583*. Proteon, Inc., USA, 1994.
- [10] R. Coltun; Fuller, V. *The OSPF NSSA Option, RFC 1587*. RainbowBridge Communications, Stanford University, 1994.