ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

CREACIÓN DE SOFTWARE DE ANÁLISIS ESTADÍSTICO DEL TRÁFICO DE INTERNET APLICABLE A UNA RED DE ÁREA LOCAL

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

ORTEGA ÁLVAREZ, GALO EFRÉN
galoefren@hotmail.com
VELASCO RIVERA, SAULO ISMAEL
saulo_velasco@hotmail.com

DIRECTOR: Ing. Xavier Calderón
acalderon@mailfie.epn.edu.ec

Quito, Octubre de 2010

DECLARACIÓN

Nosotros, Ortega Álvarez Galo Efrén y Velasco Rivera Saulo Ismael, declaramos bajo juramento que el trabajo aquí descrito es de nuestra autoría; que no ha sido previamente presentada para ningún grado o calificación profesional; y, que hemos consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedemos nuestros derechos de propiedad intelectual correspondientes a este trabajo, a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Ortega Álvarez Galo Efrén

Velasco Rivera Saulo Ismael

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Ortega Álvarez Galo Efrén y Velasco Rivera Saulo Ismael, bajo mi supervisión.

Ing. Xavier calderón
DIRECTOR DE PROYECTO

AGRADECIMIENTOS

Galo Ortega

A Dios por haberme dado la oportunidad de vivir y estar con las personas que más quiero como es mi familia.

A mis padres, quienes a lo largo de mi vida han velado por mi bienestar y educación siendo mi apoyo en todo momento. Enseñándome los valores fundamentales que forman parte de un buen ser humano. Depositando su confianza en cada reto de mi vida sin dudar ni un solo momento en mi inteligencia y capacidad.

A mis hermanas, que siempre han estado ahí pendientes en cada paso de mi vida. A Marthi, que desde pequeño supo cuidarme y junto a mis padres darme el más grande regalo que es la oportunidad de salir adelante.

A mi buen amigo Saulo, que siempre ha desempañado su papel en cada proyecto con responsabilidad y haber demostrado en cada paso una buena actitud de compañerismo y amistad.

A la Escuela Politécnica Nacional, por haber sido uno de los pilares fundamentales en mi formación universitaria.

Saulo Velasco

A mis padres por su incondicional apoyo durante todos estos años de estudio, a mis hermanos y en general a toda mi familia.

A Galo y su familia por su amistad y paciencia en el tiempo que pasamos involucrados en el desarrollo del presente proyecto de titulación.

Finalmente agradezco a la Escuela Politécnica Nacional por abrirme sus puertas y brindarme la oportunidad de conocer a tan buenos profesionales y a excelentes compañeros.

DEDICATORIA

Galo Ortega

Dedico este proyecto de titulación a Dios y a mis padres. A Dios porque siempre supo darme la energía y el descanso espiritual necesario para poder afrontar todos los obstáculos que la vida me ha deparado. A mis padres, porque han sido siempre mi ejemplo a seguir demostrando su tenacidad ante las adversidades de la vida.

Saulo Velasco

A mis padres.

CONTENIDO

ÍNDICE DE GRÁFICOS

ÍNDICE DE ECUACIONES

ÍNDICE DE TABLAS

PRESENTACIÓN

El servicio de Internet es uno de los puntos más críticos en un entorno corporativo, donde el uso óptimo de los recursos constituye el pilar fundamental para el crecimiento y estabilidad de una empresa.

Los encargados de regular el uso óptimo de este servicio son los administradores de red, siguiendo las normas de las políticas internas de la empresa.

El administrador de red debe hacer uso de herramientas de software para monitorear el ancho de banda de la conexión a Internet y también de la captura de paquetes. El problema surge al momento de la interpretación de los datos para la obtención de parámetros cuantitativos y cualitativos, en los cuales basar sus decisiones para establecer medidas restrictivas y correctivas.

El presente proyecto de titulación aborda este problema y se implementa como una solución la creación de un software con las siguientes directrices principales:

- Adquisición de datos basado en captura de paquetes.

- Almacenamiento de los valores capturados en una base de datos.

- Diferenciación de tráfico.

- Análisis de Estadística Descriptiva.

- Generación de gráficos y resúmenes de datos.

Estas características engloban los requerimientos que un software de este tipo debe poseer para tener una visión global del uso del servicio de Internet en una red local.

RESUMEN

En el primer capítulo se explora el fundamento teórico necesario con los conceptos y criterios más relevantes que aportan al proyecto de titulación. Esto incluye a la Estadística Descriptiva con sus conceptos y resúmenes gráficos, la Estadística Inferencial con una introducción a las series de tiempo, la descripción y estructura de los protocolos del modelo de referencia TCP/IP, información referente a las bibliotecas y el API (Interfaz de programación de aplicaciones) de captura de paquetes y la plataforma de desarrollo software de Java.

Posteriormente en el segundo capítulo se redacta un documento de requerimientos de software desde el punto de vista de un usuario con el perfil de un administrador de red y lo que específicamente se espera de este.

El tercer capítulo involucra a todo el desarrollo de software. Aquí se describe el diseño de la base de datos con sus tablas y campos. Por tratarse de un software desarrollado con programación orientada a objetos se incluyen los diagramas UML de casos de uso, de clases y de actividad que ayudan a describir detalladamente las relaciones y procesos principales del resultado final.

El proyecto consta de dos módulos bien diferenciados, uno especializado en la captura de paquetes y la funcionalidad de sniffer y otro enfocado al tratamiento de los datos de la base, la generación de gráficos y resúmenes, tanto visuales como en formato de texto.

En el cuarto capítulo se detallan los paquetes ejecutables finales, los requerimientos de software y de hardware. Además se describen las pruebas realizadas, el análisis de resultados, algunos escenarios de detección de anomalías y un estimado del costo de desarrollo del proyecto.

En el anexo A se describe el lenguaje de modelado UML versión 2.2 utilizado en el desarrollo del proyecto de titulación, mostrando sus aspectos esenciales como son los casos de uso, diagramas de clases, diagramas de actividad y diagramas de interacción.

El anexo B muestra los parámetros usados para acceder a la información existente en un servidor DNS (Domain Name System) utilizando el JNDI (Java Naming Directory Interface).

El anexo C se muestra los requerimientos del sistema, detalles de la instalación y ejecución del proyecto de titulación para ciertas distribuciones de CentOS 5.2, Ubuntu 8.04 y Windows (XP o superior).

El anexo D realiza una descripción comparativa del proyecto de titulación con la aplicación Wireshark 1.2.8 y Colasoft Capsa 7.1 Demo para destacar las características principales del mismo y sus limitaciones.

El anexo E incluye el estándar IEEE 830-1998 del formato de Especificación de Requisitos de Software (ERS).

El anexo F contiene la propuesta del proyecto de titulación necesaria para describir los requerimientos de software.

CAPÍTULO 1

1. INTRODUCCIÓN

En este capítulo se mostrarán los temas base para el desarrollo del presente proyecto, asumiendo que el lector posee conocimientos básicos de redes de datos, programación orientada a objetos y base de datos.

Se explorará los conceptos y términos más importantes de la Estadística Descriptiva y lo que se utilizará de la Estadística Inferencial, además de una revisión de los Protocolos del Modelo de Referencia TCP/IP y de la estructura de sus cabeceras que contienen la información pertinente a la transmisión de datos. Adicionalmente se revisarán los aspectos claves para el entendimiento de la estructura básica y funcionamiento del API de programación orientado a captura de paquetes y análisis de red.

Finalmente se mostrará información sobre las herramientas de desarrollo de software, gestor de base de datos y biblioteca de componentes estadísticos, utilizados en el desarrollo del proyecto de titulación.

1.1. ESTADÍSTICA DESCRIPTIVA [1] [2]

La necesidad de poder describir los patrones a partir de los datos provenientes del objeto de estudio, ha obligado a desarrollar un método formal de análisis de datos para el desarrollo científico y de ingeniería. El estudio de dichos datos involucra un análisis para la obtención de conclusiones, a partir de lecturas provenientes de mediciones experimentales.

La recopilación de datos es un primer paso para el análisis en la obtención de conclusiones y esta se la realiza de manera metódica con la finalidad de obtener las características más importantes de las mismas. Estas características son las más relevantes y reflejan de forma global al conjunto que representan. El

---

[1] NAVIDI, William ; Estadística para ingenieros; Cap. 1
[2] SPIEGEL, Murray y STEPHENS, Larry; Estadística; Cap. 1

conglomerado de métodos y técnicas que describe como hacer lo anteriormente expuesto, se le conoce como *estadística descriptiva*.

Los científicos y analistas se enfrentan a la perenne existencia de un rango de incertidumbre en los datos obtenidos. Los resultados pueden ser influenciados por causas aleatorias e impredecibles. Por lo tanto el objetivo de estos métodos de tratamiento de datos es el de minimizar el efecto que tienen estos factores aleatorios en las conclusiones resultantes. Este campo de estudio es el conocido como *Estadística Inferencial*.

En el presente proyecto de titulación se hará énfasis a la estadística descriptiva para el análisis de los datos obtenidos durante el proceso de captura de tráfico.

1.1.1. CONCEPTOS BÁSICOS

Para comprender con claridad el tratamiento estadístico realizado sobre los datos obtenidos por medio de la utilización del software, se describirán brevemente algunas definiciones utilizadas, referente a la estadística descriptiva.

1.1.1.1. Población

En conceptos estadísticos una población representa una colección o un conjunto completo de elementos o resultados obtenidos luego de un proceso experimental u otro evento.

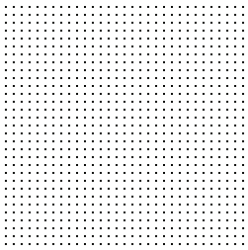


Gráfico 1.1 Población

1.1.1.2. Muestra

"Representa un subconjunto de una población, que contiene elementos o resultados que realmente se observan."[3]

Esta muestra puede o no ser representativa de una población, esto depende del método de muestreo utilizado para obtenerla.

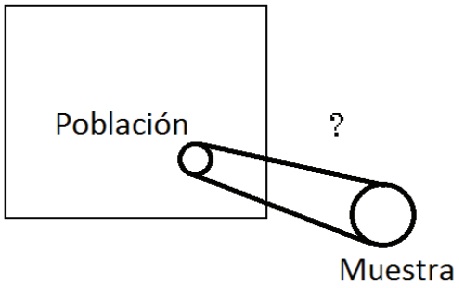


Gráfico 1.2 Muestra que representa a una Población.

1.1.1.2.1. Muestra aleatoria simple

Hace referencia a la muestra obtenida por medio de una selección aleatoria simple.

Esto implica que todos los elementos de la población tienen la misma probabilidad de ser parte de la muestra. El método consiste en asignar un número entero a cada elemento de una población de tamaño **n** en el rango que va desde **1** a **n**, posterior a esto, se deberá generar un conjunto de números aleatorios que corresponderán a los elementos que formarán parte de la muestra, como si se tratara de una lotería.

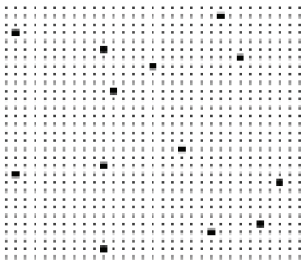


Gráfico 1.3 Muestra Aleatoria Simple

[3] Tomado de: Estadística para ingenieros, William Navidi, McGraw-Hill, pág. 3.

1.1.1.2.2. Muestra de conveniencia

Se refiere a un método por el cual se opta cuando no es posible obtener la muestra de manera completamente aleatoria.

Esto puede darse en casos donde sea físicamente impráctico o no viable la toma de una muestra para describir una población.

Como ejemplo, el caso de una fábrica de postes, en la cual elegir aleatoriamente uno de ellos dentro de una remesa apilada en grandes grupos, no resulta ser una opción práctica ya sea por disposición física u otra razón, por ello simplemente se puede escoger los postes más accesibles de la remesa.

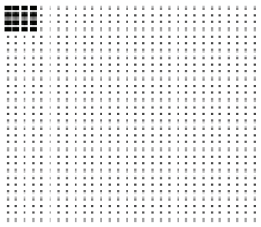


Gráfico 1.4 Muestra de conveniencia

1.1.1.3. Independencia

Cuando se habla de elementos independientes en una muestra se quiere decir que, si se conoce el valor de alguno de ellos, esto no aportaría a predecir el valor de los otros elementos restantes.

En una población finita los elementos en una muestra aleatoria simple no serían estrictamente independientes, ya que cuando se extrae un elemento de la muestra, su población cambia.

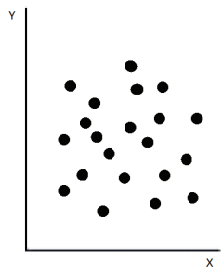


Gráfico 1.5 Independencia

1.1.2. MEDIDAS DE TENDENCIA CENTRAL

Estas medidas o promedios son conocidos de esta forma porque sus valores tienden o asemejan a los del centro del conjunto de datos, previa una ordenación por magnitud.

Entre las más utilizadas tenemos la media muestral, la mediana muestral y la moda.

Los cuartiles, deciles y percentiles pueden considerarse una extensión de la mediana.

1.1.2.1. Media muestral

La media muestral, o también conocida como media aritmética, es un valor numérico obtenido a partir de la suma de todos los elementos de la muestra, dividido para la cantidad total de elementos existentes en esta.
Se define por

$$\overline{X} = \frac{X_1 + X_2 + X_3 + \cdots + X_N}{N} = \frac{\sum_{j=1}^{N} X_j}{N} = \frac{\sum X}{N}$$

Ecuación 1.1

1.1.2.1.1. Datos atípicos

Este tipo de datos son aquellos que en ocasiones se presentan en las muestras con valores muy superiores o muy por debajo de los demás.
Pueden deberse a errores al momento del ingreso de los datos, pero también existen casos donde las poblaciones realmente contengan estos valores. Si se comprueba que realmente son fruto de un error, deben ser eliminados o corregidos, de no ser así se corre el riesgo de disminuir el carácter representativo de la muestra.

Gráfico 1.6 Conjunto de datos que contiene un dato Atípico

1.1.2.1.2. Media recortada

La media recortada es una medida de tendencia central pensada para evitar que el resultado sea influenciado por datos atípicos. Previa una ordenación de los valores de la muestra, se procede a realizar un “recorte” de un porcentaje del número de datos en ambos extremos de la muestra y se calcula la media con los elementos restantes.

Por lo general estos valores corresponden al 5%, 10% y 20% del total de elementos de la muestra.

1.1.2.2. Mediana muestral

La mediana muestral es una medida de tendencia central que corresponde al valor central (en caso de que el número de elementos de la muestra sea impar) o al promedio de los dos valores centrales (en caso de que el número de elementos de la muestra sea par) de un conjunto ordenado de números de forma creciente o decreciente.

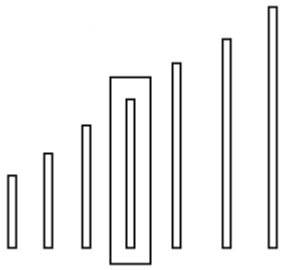


Gráfico 1.7 Mediana

1.1.2.3. Moda

La moda muestral corresponde al o los valores con mayor frecuencia en una muestra. En otras palabras, al o los valores que más se repiten.

Una muestra puede no tener una moda, o poseer varias (bimodal, multimodal).

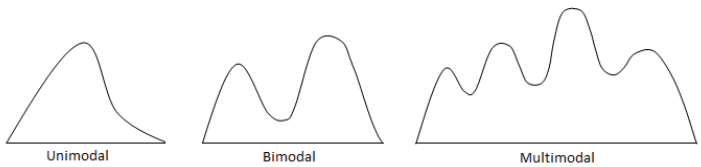


Gráfico 1.8 Modas

1.1.2.4. Cuartiles, Deciles y Percentiles

Estos valores tienen como objetivo dividir en partes iguales un conjunto de datos ordenados por magnitud. Entre los principales tenemos:

Cuartiles: dividen la muestra en cuatro partes iguales. Estos valores se denotan como Q_1, Q_2 y Q_3 que representan al primero, segundo y tercer cuartiles, donde Q_2 es igual a la mediana.

Deciles: dividen la muestra en 10 partes iguales, se indican con D_1, D_2,…, D_9.

Percentiles: dividen la muestra en 100 partes iguales, se denotan por P_1, P_2,…, P_{99}.

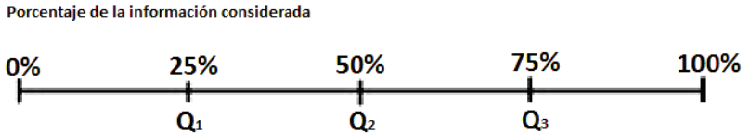


Gráfico 1.9 Representación de los cuartiles

1.1.3. MEDIDAS DE DISPERSIÓN

Se llama dispersión al grado en que los diferentes valores numéricos de los datos tienden a extenderse alrededor del valor medio utilizado dentro de un conjunto de datos y permite retratar la distancia de los valores o la concentración de los datos en cierto sector o recorrido de la variable.

Es importante conocer si los valores en general están cerca o alejados de valores centrales, es por ello que el grado de dispersión se mide por medio de indicadores estadísticos denominados medidas de dispersión.

1.1.3.1. Rango

Se define como la diferencia entre los dos valores extremos que toma un conjunto de números. Es la medida de dispersión más sencilla y la que proporciona menos información.

Cabe decirse, por otra parte, que esta medida presenta una serie de inconvenientes, pues el hecho de que no influyan más de dos valores del total de la serie puede provocar una deformación de la realidad. Al mismo tiempo puede

verse afectada por la presencia de ciertos valores extremos que son poco representativos.

1.1.3.2. Desviación media

La desviación media viene a indicar el grado de concentración o de dispersión de los valores de la variable. Se suele centrar en la medida de la desviación con respecto a la media. Si es muy alta, indica gran dispersión; si es muy baja refleja un buen agrupamiento y que los valores son parecidos entre sí.

La desviación media se puede utilizar como medida de dispersión en todas aquellas distribuciones en las que la medida de tendencia central más significativas haya sido la media.

Puede definirse como la media aritmética de las desviaciones de cada uno de los valores con respecto a la media aritmética de la distribución, y se indica así:

$$DM = \frac{\sum_{j=1}^{N} |X_j - \overline{X}|}{N} = \frac{\sum |X - \overline{X}|}{N}$$

Ecuación 1.2

La desviación media en el caso de datos agrupados en intervalos viene dado como:

$$DM = \frac{\sum_{j=1}^{K} f_j |X_j - \overline{X}|}{N} = \frac{\sum f |X - \overline{X}|}{N}$$

Ecuación 1.3

Para valores de X_1, X_2,…, X_K que ocurren con frecuencias f_1, f_2,…, f_K respectivamente, donde $N = \sum_{j=1}^{K} f_j = f$.

En la fórmula anterior las desviaciones van multiplicadas por las frecuencias de clase de los intervalos correspondientes. Las desviaciones son de cada centro o marca de clase representado por X_j respecto a la media aritmética.

1.1.3.3. Desviación estándar

La desviación estándar nos da como resultado un valor numérico que representa la media cuadrática o promedio de diferencia que hay entre los datos y la media.

La desviación estándar nos informa sobre la dispersión de los datos respecto al valor de la media; cuanto mayor sea su valor, más dispersos estarán los datos.

Se define como:

$$s = \sqrt{\frac{\sum_{j=1}^{N}(X_j - \overline{X})^2}{N}} = \sqrt{\frac{\sum(X - \overline{X})^2}{N}}$$

Ecuación 1.4

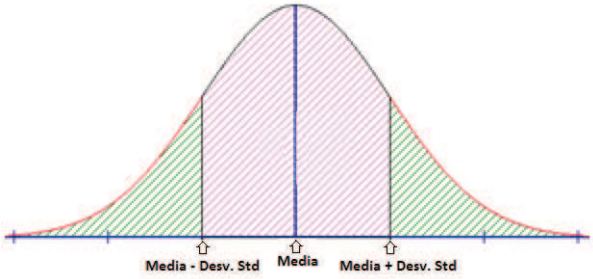


Gráfico 1.10 Desviación estándar

1.1.3.4. Varianza muestral

La varianza muestral se define como la media de los cuadrados de las variaciones a la media muestral.

Es una medida de dispersión utilizada en la estadística que nos indica cuanto distan las observaciones de una variable con respecto al promedio.

La varianza muestral esta denotada por s^2, de los datos X_j,.. X_N y por sigma σ^2 para la varianza poblacional.

$$s^2 = \frac{\sum_{j=1}^{N}(X_j - \overline{X})^2}{N} = \frac{\sum(X - \overline{X})^2}{N}$$

Ecuación 1.5

1.1.4. RESÚMENES GRÁFICOS

Una gráfica o diagrama es una representación complementaria a una tabla o cuadro, que permite observar las tendencias de un comportamiento en estudio y facilita el análisis estadístico de las variables allí relacionadas.

1.1.4.1. Clases o Categorías

En la etapa de organización de los datos, se hace referencia a la clasificación y tabulación de los mismos. Cuando se tienen grandes cantidades de datos es necesario dividir la información en clases o categorías. Un elemento cualquiera del conjunto de datos pertenecerá a una clase determinada, si se encuentra en el

rango que fue definido para esa clase. Por ejemplo para una muestra de bosque amazónico de 100 metros cuadrados, la altura comprendida entre 10 y 15 metros corresponde a un rango de datos y cualquier árbol con una altura dentro esos valores, pertenecerá a esa clase o categoría.

Otros ejemplos útiles pueden ser: los estudiantes de un establecimiento educativo cuya edad esté entre los 15 y 17 años, la tasa de transferencia de Internet en el intervalo de 20 a 30 KBps, etc.

1.1.4.2. Tabla de frecuencias

En la estadística descriptiva el principal objetivo es representar conjuntos de datos mediante tablas o gráficos resumen, con el fin de poder identificar el comportamiento característico de un fenómeno y facilitar su análisis exhaustivo.

Cualquier análisis que se emprenda puede conducir a la acumulación de valores cuantitativos y cuasi-cualitativos correspondientes a las diversas medidas efectuadas. Esta posibilidad, convierte a la estadística en una herramienta vital para el tratamiento de volúmenes de datos mediante tablas resúmenes conocidas como Tabla de frecuencias.

De esta forma la Tabla de frecuencias o distribución de frecuencias es una ordenación en forma de tabla de los datos estadísticos, asignando a cada dato su frecuencia correspondiente. Cuando los datos son agrupados en clases, la interpretación resulta ser más sencilla.

1.1.4.2.1. Frecuencia absoluta.

Es la cantidad de datos que integran cada una de las clases, es decir, el número de valores que encontramos dentro de un mismo intervalo de clase o categoría.

1.1.4.2.2. Frecuencia relativa.

Es la cantidad de repeticiones obtenidas para cada clase de datos o categoría, en relación al total de las observaciones. Resulta de dividir la cantidad de elementos de cada clase (frecuencia absoluta) por el tamaño de la muestra.

Las frecuencias relativas son un porcentaje, ya que relacionan una parte del conjunto con el total.

1.1.4.2.3. Frecuencias acumuladas.

La frecuencia acumulada de un intervalo de clase es aquella que suma las frecuencias anteriores hasta dicho intervalo.

La frecuencia relativa acumulada tiene una definición igual a la anterior, pero esta usa las frecuencias relativas.

En el siguiente ejemplo se describe los diferentes tipos de frecuencias antes mencionados. Para ello se asume que un dado que fue lanzado 100 veces, el resultado de dicho evento se muestra en la tabla 1.1 que contiene los diferentes valores de frecuencias para cada cara del dado.

Caras del dado	Frecuencia absoluta (número de lanzamientos del dado)	Frecuencia relativa	Frecuencia relativa acumulada
1	11	0.11	0.11
2	13	0.13	0.24
3	15	0.15	0.39
4	18	0.18	0.57
5	18	0.18	0.75
6	25	0.25	1.00

Tabla 1.1 Evento del lanzamiento de un dado (Tabla de frecuencias).

1.1.4.3. Histogramas

Es una representación gráfica de las distribuciones de frecuencias por medio de barras o rectángulos para cada intervalo de clase, dónde la superficie de cada barra es proporcional a la frecuencia de sus valores representados.

En el eje vertical se representan las frecuencias y en el eje horizontal los valores de la variable, normalmente señalando las marcas de clase, es decir, la mitad del intervalo en el que están agrupados los datos.

Las principales características de los histogramas que aportan al análisis de datos son:

Síntesis: Permitir resumir grandes cantidades de datos.

Análisis: Permite el análisis de los datos mostrando esquemas de comportamiento y pautas de variación que tienen una gran complejidad de interpretación en una tabla numérica.

Capacidad de Comunicación: Permite comunicar información de forma clara y sencilla sobre situaciones complejas.

1.1.4.3.1. Tipos de Histogramas

Los histogramas se dividen en:

Diagrama de barras simples

Este tipo de diagrama representa la información de una variable. La frecuencia que puede ser absoluta o relativa se representa mediante la altura de la barra, la cual es proporcional a la frecuencia del intervalo de clase correspondiente.

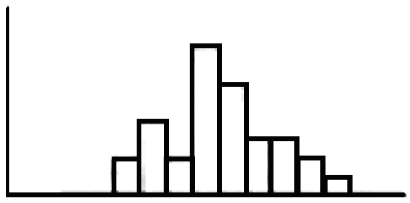


Gráfico 1.11 Histograma de barras simples

Diagrama de barras compuesta

Estos diagramas permiten representar la información de una tabla de entrada doble (2 variables). La altura de la barra representa la frecuencia de la clase y esta altura es proporcional a la frecuencia de cada una de ellas.

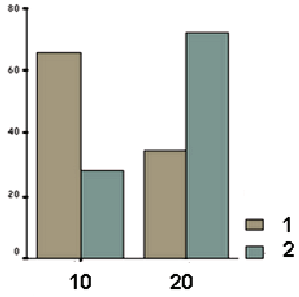


Gráfico 1.12 Histogramas barras compuestas

Diagramas de barra agrupada

Esta es utilizada para representar la información de una tabla de doble entrada o más, es decir, a partir de dos variables. La representación se la hace mediante un conjunto de barras correspondientes a cada clase o categoría que se clasifican respecto a las diferentes variables.

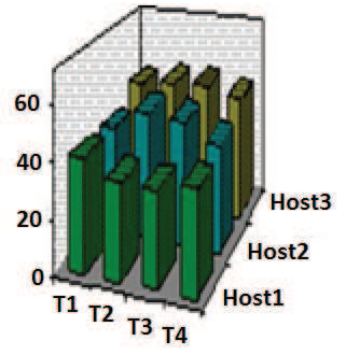


Gráfico 1.13 Histogramas barras Agrupadas

Polígono de frecuencias

Este es un gráfico de líneas de las frecuencias de clase de una distribución, en el cual la altura del punto asociado a una clase es proporcional a la frecuencia de esta. Se lo obtiene uniendo los puntos medios de la parte superior de los bloques o rectángulos del histograma.

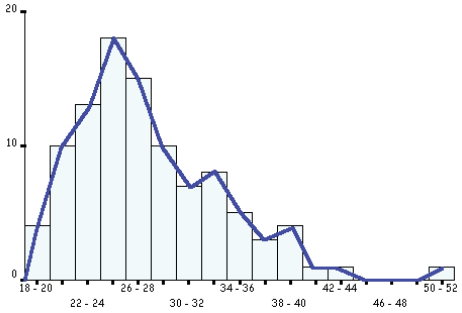


Gráfico 1.14 Polígono de Frecuencias

Ojiva Porcentual o Polígono de Frecuencias Relativas Acumuladas

Es un gráfico que recoge las frecuencias relativas acumuladas para cada intervalo de clase (frecuencia acumulada dividida entre la frecuencia total). Es muy útil cuando se requiere representar un rango porcentual de cada clase en una distribución de frecuencias.

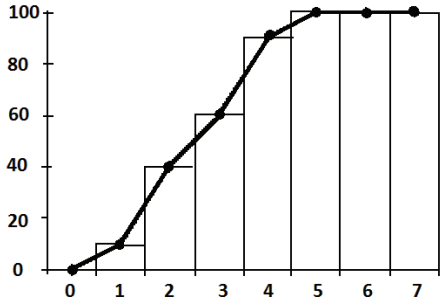


Gráfico 1.15 Ojiva Porcentual

1.2. ESTADÍSTICA INFERENCIAL

Es una parte de la estadística que comprenden métodos y procedimientos, mediante los cuales se deducen (infieren) propiedades o características de una población a partir de una muestra significativa.

Una aplicación avanzada de la Estadística Inferencial es la obtención de expresiones matemáticas para las series de tiempo, con curvas de tendencia con el objetivo de realizar estimación o predicción.

1.2.1. SERIES DE TIEMPO

Una serie de tiempo es un conjunto de datos ordenados provenientes de observaciones realizadas a lo largo del tiempo, en momentos específicos y generalmente con intervalos de igual duración.

Matemáticamente una serie de tiempo se define como el conjunto de valores de una variable dependiente Y correspondiente a los tiempos t_1, t_2,..., es decir, Y es una función de **t**, denotado por Y=F(t).

En el presente proyecto se utilizarán las series de tiempo solo en su representación gráfica, a partir de los datos recopilados en la captura de paquetes de tráfico de Internet, lo que arroja información suficiente para la toma de decisiones de un administrador de red.

1.2.1.1. Componentes de las Series de Tiempo

Las series de tiempo pueden abstraerse como un conjunto de de movimientos característicos o variaciones, presentes en distintos grados.

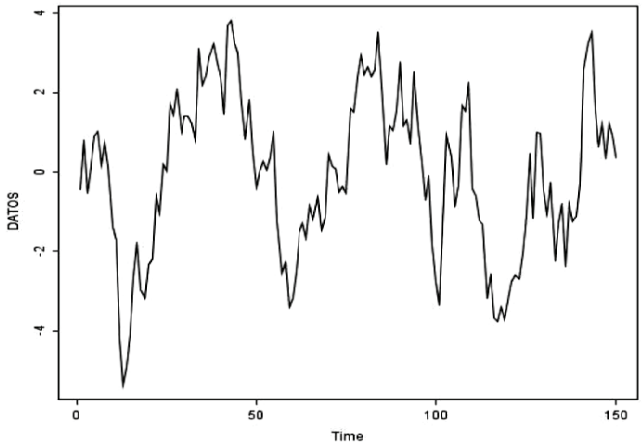


Gráfico 1.16 Series de Tiempo.

Estos movimientos característicos pueden clasificarse en cuatro tipos principales.

1. **Movimientos a largo plazo o seculares.** Llamados también *variación secular* o *tendencia secular*. Este movimiento describe la dirección general que la gráfica sigue en un intervalo grande de tiempo. Suele representarse por una *curva de tendencia* obtenida por distintos métodos.

2. **Movimientos cíclicos o variaciones cíclicas.** Se refieren a las oscilaciones o movimientos respecto de una recta o curva de tendencia. Estos *ciclos* son no periódicos, es decir, pueden o no seguir patrones.

3. **Movimientos estacionales o variaciones estacionales.** Se refiere a los patrones recurrentes casi idénticos, que ocurren en períodos iguales o inferiores a un año. Dependiendo del tipo de datos disponibles pueden extenderse para un período determinado como días, horas, semanas.

4. **Movimientos irregulares o aleatorios.** Son movimientos fortuitos, que suceden por causas aleatorias, cuyo efecto suele observarse como variaciones de corta duración. Sin embargo puede darse el caso de que por su magnitud resulten en movimientos cíclicos o de otro tipo.

1.2.1.2. Análisis de Series de Tiempo

Se refiere a la descripción matemática de los movimientos componentes presentes en una serie de tiempo.

Vista de esta forma una serie de tiempo representada por Y puede describirse como un producto de las variables T, C, S e I, responsables de los movimientos de tendencia, cíclicos, estacionales e irregulares, respectivamente.

$$Y = TxCxSxI=TCSI$$

Ecuación 1.6

Por lo tanto, el análisis de series de tiempo consiste en buscar cada uno de estos factores, es decir, la descomposición de series de tiempo en sus componentes básicos.

La serie de tiempo Y también puede considerarse como la suma de sus componentes básicos. Esto depende de las preferencias de los estadísticos.

1.3. PROTOCOLOS DEL MODELO DE REFERENCIA TCP/IP

El modelo de referencia TCP/IP, nace del resultado de una investigación realizada en una red experimental de conmutación de paquetes ARPANET, financiada por la Agencia de Proyectos de Investigación Avanzada para la Defensa (DARPA). Este modelo de referencia consiste en una extensa colección de protocolos que han surgido como estándares de Internet, siendo así la arquitectura más adoptada para la interconexión de sistemas.

La estructura fundamental de la red TCP/IP es la de un sistemas de conmutación de paquetes.

TCP/IP no posee un modelo oficial de referencia, sino que se ha basado en los protocolos estándares y tareas involucradas para la comunicación, que han permitido formar una organización de 4 capas relativamente independientes.

- Capa Aplicación.
- Capa Transporte.
- Capa Internet.
- Capa de Acceso a la Red.

1.3.1. PROTOCOLOS DE CAPA INTERNET

La función de esta capa es la de permitir intercambiar información entre dos dispositivos conectados en diferentes redes interconectadas, a través de una serie de procedimientos.

1.3.1.1. Protocolo IP

IP (Internet Protocol), se utiliza en esta capa para ofrecer el servicio de enrutamiento a través de varias redes, y se implementa tanto en sistemas finales como en dispositivos comunicación intermedios.

IP es un protocolo no confiable, no orientado a conexión, no posee un mecanismo de corrección de errores y su unidad de transferencia de datos es el datagrama. Además intercambia mensajes de error y de control entre nodos a través de ICMP (Internet Control Message Protocol).

La base del modelo TCP/IP nace de las características de IP ya que el servicio de entrega de datagramas es sin conexión, poco confiable, y del mejor esfuerzo.

1.3.1.2. Direccionamiento IP

El protocolo IP ha sido el fundamento de Internet y virtualmente de todas las redes privadas, por ello este tipo de protocolo maneja un direccionamiento para poder identificar cada host y dispositivos de conmutación dentro de una Intranet como de Internet.

Cabe mencionar que IP no identifica un dispositivo de conmutación o un host, sino un interfaz de red. En un caso particular un host puede tener tantas direcciones IP como dispositivos de red tenga.

Una dirección IP está conformada por 32 bits, agrupada en 4 números decimales separados por puntos. Esta estructura está compuesta por dos partes; número de red, número de host.

El número de red identifica a una red específica, y el número de host identifica a una estación de trabajo dentro de esa red.

El protocolo IP maneja una cabecera para formar los diferentes datagramas para el intercambio de información, detallado en el gráfico 1.17.

bit	0	4	8	16	19 31
20 octetos	Versión	IHL	Tipo de Servicio	Longitud Total	
	Identificación			Indicadores	Desplazamiento de fragmento
	Tiempo de Vida	Protocolo		Suma de Comprobación de Cabecera	
	Dirección Origen				
	Dirección Destino				
	Opciones + Relleno				

Gráfico 1.17 Estructura del Paquete IP [4]

Versión (4 bits): Indica el número de la versión del protocolo, con el cual se está trabajado.

Longitud de la cabecera Internet (IHL, Internet Header Length) (4 bits): Indica la longitud de la cabecera expresada en palabras de 32 bits, con su valor mínimo de 5 correspondiente a la longitud mínima de de cabecera de 20 octetos.

Tipo de servicio (8 bits): especifica los parámetros de seguridad, prioridad, retardo y rendimiento.

Longitud Total (16 bits): longitud total del datagrama, en octetos.

Identificador (16 bits): Es un número de secuencia que permite identificar de forma única al datagrama.

Identificadores (3 bits): Estos identificadores permiten setear opciones para segmentación y reensamblado de datagramas.

Desplazamiento del Fragmento (13 bits): Indica el lugar donde se sitúa el fragmento dentro del datagrama original, medido en unidades de 64 bits.

Tiempo de Vida (8 bits): Especifica cuanto tiempo, en segundos, se le permite al datagrama permanecer en la red.

Suma de comprobación de cabecera (16 bits): Es un código de detección de errores basado en la suma de complementos solo aplicado a la cabecera, y este valor se re calcula en cada dispositivo de encaminamiento.

---

[4] Fuente: W. Stallings - Comunicaciones y Redes de Computadores (6ª Edición)

Dirección de origen (32 bits): Especifica la dirección origen de donde proviene el datagrama IP.

Dirección de destino (32 bits): Específica la dirección destino a de donde se dirige el datagrama IP.

Opciones (variable): Contiene las opciones solicitadas por el usuario que envía los datos.

Relleno (variable): Se usa para asegurar que la cabecera del datagrama tenga una longitud múltiplo de 32 bits.

Datos (variable): El campo de datos debe tener una longitud múltiplo de 8 bits. La longitud máxima de todo el datagrama con cabecera podría llegar hasta 65535 octetos.

1.3.1.2.1. Clases de direcciones IP

La dirección IP, permite una asignación variable de bits, para especificar la red y el computador. Además permite flexibilidad para asignar direcciones a los computadores correspondientes a redes con diferentes tamaños y se subdividen en 5 clases.

Clase A: Pocas redes, cada una con muchos hosts.

Clase B: Un número medio de redes, cada una con un número medio de hosts.

Clase C: Muchas redes, cada una con pocos hosts.

32 bits									
Bit	1	1	1	1	4	8	8	8	Rango de direcciones de host
Clase	0	Red				Host			1.0.0.0 to 127.255.255.255
A	1	0	Red				Host		128.0.0.0 to 191.255.255.255
B	1	1	0	Red				Host	192.0.0.0 to 223.255.255.255
C	1	1	1	0	Dirección Multicast				224.0.0.0 to 239.255.255.255
D	1	1	1	1	Reservado para uso futuro				240.0.0.0 to 255.255.255.255

Gráfico 1.18 Clases de direcciones IP

La tabla 1.2 resume todas las direcciones IP con sus rangos válidos.

Clase de Red	Nº de bits subred/Host	Direcciones de Subred (Recordar que el rango que abarca la subred depende del MS)	Nº Máquinas por subred que son numerables	Rango de direcciones IP Rango válido (RV)
A	8/24 255.0.0.0	0.0.0.0 … 127.0.0.0	(2^24)-2 = 16777214	**RV** 1.0.0.1 … 127.255.255.254
B	16/16 255.255.0.0	128.0.0.0 … 191.255.0.0	(2^16)-2 = 65534	**RV** 128.0.0.1 … 191.255.255.254
C	24/8 255.255.255.0	192.0.0.0 … 223.255.255.0	(2^8)-2 = 254	**RV** 192.0.0.1 … 223.255.255.254
D	32/0 255.255.255.255	…	Multicast, Fut Resev	224.0.0.0 … 239.255.255.255
E	32/0 255.255.255.256	…	Multicast, Fut Resev	240.0.0.0 … 255.255.255.255

Tabla 1.2 Direcciones IP.

1.3.1.2.2. Direcciones reservadas para intranet

Por la amplia demanda de direcciones IP para la salida a Internet (direcciones públicas), se ha determinado un grupo de direcciones IPv4 como direcciones reservadas para Intranet (direcciones privadas), para de esta manera poder optimizar el uso de direcciones IP. Este tipo de direcciones se las usa en cada red privada sin causar conflicto al momento de salir a Internet.

Existen varios rangos de direcciones reservadas para cada clase de direcciones IP.

Clase de Red	Rango de direcciones IP	
	Desde	Hasta
Clase A	10.0.0.0	10.255.255.255
Clase B	172.16.0.0	172.31.255.255
Clase C	192.168.0.0	192.168.255.255

Tabla 1.3 Direcciones IP Privadas.

1.3.2. PROTOCOLOS DE CAPA TRANSPORTE

Los protocolos de capa transporte tienen como finalidad la comunicación extremo-a-extremo entre dos entidades distintas. Estos protocolos hacen transparente el proceso de transferencia de datos a todas las capas superiores, es decir, que no necesitan conocer detalles de la red o redes en los que se desarrolla la comunicación.

Los protocolos de capa transporte dentro de la arquitectura de protocolos TCP/IP se ubican sobre la capa de red y debajo de la capa de aplicaciones. Estos proporcionan servicios a los protocolos de capa aplicación (FTP, SMTP, POP3, etc.) y hacen uso de los servicios de algún protocolo de la capa inferior para establecer una comunicación entre la entidad local y la remota.

Existen dos tipos de servicio de transporte: orientado a la conexión y no orientado a la conexión.

TCP/IP consta de dos protocolos de capa transporte: *Transmission Control Protocol* (TCP) o protocolo de control de transmisión y el *User Datagram Protocol* (UDP) o protocolo de datagrama de usuario. El primero es orientado a la conexión y el segundo no orientado a la conexión.

1.3.2.1. Protocolo TCP

"*Un servicio orientado a la conexión proporciona el establecimiento, mantenimiento y cierre de una conexión lógica entre usuarios TS (Transport Service)*". [5]

Este protocolo de capa transporte (especificado en el RFC 793) permite la comunicación segura entre dos procesos usuarios de este servicio con mecanismos que permiten independencia del servicio de interconexión de red utilizado, sea este seguro o no seguro, sobre una sola red o a través de varias interconectadas.

La unidad de datos de este protocolo es llamado segmento TCP. Dada la complejidad de los mecanismos de TCP, su cabecera es relativamente grande y contiene los siguientes campos, como se muestra en el gráfico 1.19.

<table>
<tr><td>bits</td><td colspan="3">0</td><td colspan="3">4</td><td>10</td><td>11</td><td>12</td><td>13</td><td>14</td><td>15</td><td colspan="2">16</td><td>31</td></tr>
<tr><td rowspan="7"></td><td colspan="9">Número de Puerto Origen</td><td colspan="6">Número de Puerto Destino</td></tr>
<tr><td colspan="15">Número de Secuencia</td></tr>
<tr><td colspan="15">Número de Asentimiento</td></tr>
<tr><td colspan="3">Longitud de cabecera</td><td colspan="3">Reservado(6 bits)</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td colspan="2">Tamaño de ventana</td></tr>
<tr><td colspan="9">Checksum de todo el segmento TCP</td><td colspan="6">Puntero de datos urgentes</td></tr>
<tr><td colspan="15">Opciones (en su caso)</td></tr>
<tr><td colspan="15">Datos (en su caso)</td></tr>
</table>

Gráfico 1.19 Estructura del Segmento TCP.

[5]Tomado de: STALLINGS, William; Comunicaciones y Redes de Computadores (6ª Edición) ; Cap. 17 ; pág. 566

Número de puerto origen (16 bits): puerto TCP origen.

Número de puerto destino (16 bits): puerto TCP destino.

Número de secuencia (32 bits): número de secuencia del primer octeto del campo de datos en este segmento.

Número de asentimiento o confirmación (32 bits): es un número llamado también *acuse de recibo*. Este valor contiene el número de secuencia del siguiente octeto de datos que la entidad TCP destino espera recibir.

Longitud de la cabecera (4 bits): indica el número de palabras de 32 bits contenidos en la cabecera.

Reservados (6 bits): campo de bits reservados para uso futuro.

Indicadores: grupo de 6 bits con significado independiente.

URG: cuando tiene un valor igual a 1, indica que el campo *puntero de datos urgentes* es válido.

ACK: cuando es igual a 1, indica que el campo de confirmación o acuse de recibo es válido.

PSH: indica la función de carga de TCP, es decir, se transmiten los datos sin que se hayan acumulado los suficientes para formar un segmento.

RST: cierre abrupto de la conexión por parte de un usuario TCP.

SYN: utilizado para sincronizar los números de secuencia cuando se establecen las conexiones.

FIN: es usado para el cierre ordenado de la conexión y es enviado en el segmento que contiene los últimos datos.

Tamaño de la ventana (16 bits): Contiene el número de bytes que el que envía está en capacidad de aceptar.

Checksum o suma de verificación (16 bits): es el complemento a 1 de la suma modulo 2^{16} -1 de todas las palabras de 16 bits presentes.

La suma de verificación se realiza sobre todo el segmento TCP más una seudo-cabecera, con la finalidad de inspeccionar errores en la trasmisión de los datos y asegurar que el segmento TCP llegó a su destino correspondiente. Esta seudo-cabecera está compuesta por los siguientes campos de la cabecera IP: dirección IP de origen y destino, el protocolo más un campo de longitud de segmento TCP.

bits	0 8	16	31
	Dirección IP origen		
	Dirección IP destino		
	Zero	Protocolo	Longitud TCP

Gráfico 1.20 Seudo-cabecera TCP

Puntero de datos urgentes (16 bits): señala la posición del byte siguiente, correspondiente a los datos urgentes.

Opciones (Variable): permite especificar el tamaño máximo del segmento que podrá ser aceptado, en caso de que sea necesario.

1.3.2.2. Protocolo UDP

UDP es un protocolo de capa transporte que provee de un servicio no orientado a conexión para los procesos de capa aplicación. Se encuentra especificado en el RFC 768.

El protocolo UDP no garantiza la entrega segura de la información o de impedir duplicados, por lo tanto es un servicio no seguro. Sin embargo UDP es utilizado en situaciones donde los mecanismos de establecimiento y mantenimiento de la conexión son innecesarios o desfavorables por su naturaleza.

Algunos casos con estas características serían la recolección de datos de entrada, difusión de datos de salida en un entorno de red, transacciones petición/respuesta donde el servidor es el encargado de la gestión del servicio y las aplicaciones en tiempo real como voz o contenido multimedia donde la retransmisión sería contraproducente por el retardo que causa y por la tolerancia a errores en este tipo de conexiones.

Los mecanismos de UDP son relativamente simples en comparación a TCP, su cabecera así lo demuestra como se ve en el siguiente gráfico.

bit	0	16	31
8 Octetos	Puerto Origen		Puerto Destino
	Longitud		Suma de Comprobación

Gráfico 1.21 Cabecera UDP[6]

Número de puerto origen (16 bits): puerto UDP origen.

Número de puerto destino (16 bits): puerto UDP destino.

Longitud (16 bits): tamaño del segmento UDP en unidades de palabras de 16 bits, incluida cabecera y datos.

Suma de comprobación o verificación (16 bits): usa el mismo algoritmo que TCP. Esta es aplicada sobre todo el segmento UDP mas una seudo-cabecera igual a la utilizada en TCP. En caso de error se procede a descartar el segmento y no existe ningún procedimiento adicional.

1.3.2.3. Puertos bien conocidos (WELL KNOWN PORT NUMBERS)

Los puertos representan el destino de las conexiones lógicas usadas en conversaciones de larga duración, por protocolos como TCP y UDP.

La abstracción de puerto surgió con la necesidad de proveer servicios a usuarios desconocidos, para ello un puerto de servicio debe ser definido.

La entidad responsable de asignar estos puertos es la IANA[7] y se encuentra definido en el RFC 1700, que trata sobre todos los *Assigned Numbers* utilizados en la comunidad de Internet.

La asignación de puertos es válida tanto para TCP y UDP. La cantidad de puertos asignados, actualmente se encuentra en el rango de 0 a 1023.

---

[6] Fuente: W. Stallings - Comunicaciones y Redes de Computadores (6ª Edición)
[7] Internet Assigned Numbers Authority

A continuación se listarán algunos de los más importantes:

Keyword	Decimal	Description
echo	7/tcp	Echo
echo	7/udp	Echo
ftp-data	20/tcp	File Transfer [Default Data]
ftp-data	20/udp	File Transfer [Default Data]
ftp	21/tcp	File Transfer [Control]
ftp	21/udp	File Transfer [Control]
telnet	23/tcp	Telnet
telnet	23/udp	Telnet
smtp	25/tcp	Simple Mail Transfer
smtp	25/udp	Simple Mail Transfer
tftp	69/tcp	Trivial File Transfer
tftp	69/udp	Trivial File Transfer
www-http	80/tcp	World Wide Web HTTP
www-http	80/udp	World Wide Web HTTP
nntp	119/tcp	Network News Transfer Protocol
nntp	119/udp	Network News Transfer Protocol

1.3.3. PROTOCOLOS DE CAPA APLICACIÓN

El conjunto de protocolos TCP/IP tiene como objetivo el de brindar soporte a las distintas aplicaciones de usuario. Cada una de ellas es manejada por un módulo bien diferenciado.

Existen distintos tipos de aplicaciones, entre las más comunes se podrían citar las siguientes:

Transferencia de archivos:

- FTP (File Transfer Protocol)
- TFTP (Trivial File Transfer Protocol)

Correo electrónico:

- SMTP (Simple Mail Transfer Protocol)
- MIME (Multipurpose Internet Mail Extensions)

Acceso remoto:

- TELNET (TELecommunication NETwork)
- RLOGIN (Remote Login)

Gestión de red:

- SNMP (Simple Network Management Protocol)

Resolución de nombres de dominio:

DNS (Domain Name System)

Aplicaciones cliente-servidor que supongan la utilización de hipertexto[8]:

- HTTP (HyperText Transfer Protocol)

Los protocolos de capa aplicación hacen uso de los protocolos TCP y UDP de la capa transporte, para la comunicación extremo a extremo entre los hosts involucrados. Por ello estos protocolos tienen asociado un número de puerto correspondiente según la asignación de la IANA en el RFC 1700.

1.4. API DE PROGRAMACIÓN ORIENTADO A CAPTURA DE PAQUETES Y ANÁLISIS DE RED

Con el actual desarrollo de las redes de comunicaciones se ha vuelto una necesidad la creación de aplicaciones personalizadas que permitan el acceso a la red para su administración. También existen protocolos estandarizados que recogen información a través de la red y la envían a un nodo central para su interpretación, como es el caso de SNMP. Sin embargo esta información recogida se encuentra limitada a las distintas MIBs[9] habilitadas, no pudiendo profundizar en el contenido mismo del flujo de datos.

En respuesta a esta necesidad se han desarrollado APIs[10], algunos gratuitos y otros comerciales, con sus respectivas ventajas y limitaciones.

Tomando en cuenta la gran difusión de sistemas operativos Win32[11], se ha elegido para el presente proyecto de titulación a WinPcap y jNetPcap. A continuación será explicado el objetivo y funcionalidad de los mismos.

---

[8] El hipertexto es una tecnología que organiza una base de información en bloques distintos de contenidos, conectados a través de una serie de enlaces cuya activación o selección provoca la recuperación de información.
Referencia: http://www.ldc.usb.ve/~abianc/hipertexto.html.

[9] MIB (*Management Information Base*) es una base de datos que contiene información jerárquica, estructurada en forma de árbol, de todos los dispositivos gestionados en una red de comunicaciones.

[10] API *(Application Programming Interface).*

1.4.1. WINPCAP [12] [13]

Winpcap es una biblioteca de código abierto para captura de paquetes y análisis de red para sistemas operativos que utilizan la plataforma Win32.

Winpcap propone una arquitectura que añade varias características propias de sistemas Unix a la plataforma Win32. Unix provee de un conjunto de llamadas al sistema que permiten a las aplicaciones interactuar con la red directamente. Estas primitivas pueden ser usadas en aplicaciones de captura de paquetes, con el cual se toma el flujo de datos a través de la red *sin ningún procesamiento de protocolos por parte del sistema operativo*, es decir, lo que se conoce como "raw packets".

Winpcap incluye en sus capacidades las siguientes facilidades:

- Captura de "raw packets", tanto los destinados al host donde se encuentra instalada la aplicación como los intercambiados por otros hosts.
- Filtrado de paquetes de acuerdo a reglas especificadas por el usuario antes de ser enviados a la aplicación.
- Transmisión o inyección de "raw packets" en la red.
- Obtención de información estadística del tráfico de red.

La potencialidad de Winpcap puede ser aprovechada en varios tipos de herramientas de red que pueden ser de análisis, resolución de problemas, seguridad y monitorización. Más específicamente podrían ser:

- Analizadores de red y protocolos.
- Loggers[14] de tráfico.
- Generadores de tráfico.
- Bridges y routers de nivel de usuario.
- Sistemas de detección de intrusos (NIDS[15]).
- Escáneres de red.
- Herramientas de seguridad.

---

[11] Win32 es un conjunto de funciones residentes en bibliotecas generalmente dinámicas, también llamadas DLLs, que permiten ejecutar aplicaciones en un sistema operativo Windows de 32 bits.
[12] Referencia: http://www.winpcap.org/
[13] Fuente: An Architecture for High Performance Network Analysis - RISSO, Fulvio; DEGIOANNI, Loris.
[14] El Logger es un registro oficial de eventos durante un rango de tiempo en particular.
[15] Network Intrusion Detection Systems

Sin embargo WinPcap tiene limitaciones, es decir, no posee la capacidad de bloquear, filtrar o manipular el tráfico generado por otros programas en la misma máquina. Por esta razón no puede ser usado en aplicaciones como limitadores de tráfico, planificadores de QoS[16] y firewalls personales.

1.4.1.1. Módulos WinPcap

El conjunto de características se las obtiene por medio de un driver de dispositivo, el cual es instalado dentro del módulo de networking del núcleo de Win32, más un par de DLLs[17] en el nivel de usuario.

En el gráfico puede observarse la estructura básica de WinPcap:

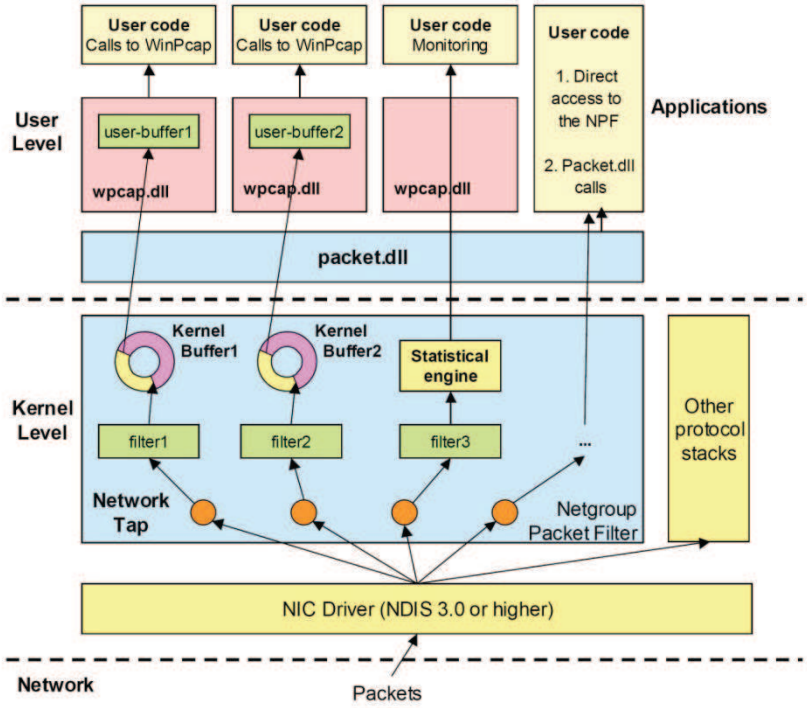


Gráfico 1.22. WinPcap y NPF. [18]

El primer módulo es parte del nivel de kernel denominado Netgroup Packet Filter (NPF) que filtra los paquetes, los entrega intactos al nivel usuario.

El segundo módulo está contenido en *packet.dll*, se encarga de proveer una interfaz común al driver de paquetes perteneciente a las plataformas Win32 sin necesidad de ser recompilada para cada versión de Windows, es decir, ofrece un API independiente del sistema. Incluye funciones de bajo nivel para obtener los

---

[16] Quality of Service
[17] Dynamic Link Libraries
[18] Fuente: Profiling and Optimization of Software-Based Network-Analysis Applications - DEGIOANNI, Loris; BALDI, Mario; RISSO, Fulvio; y VARENNI, Gianluca

nombres de los adaptadores de red del sistema, carga dinámica del driver, acceso a información como la máscara de red y algunos contadores de hardware (número de colisiones en Ethernet, etc.).

El tercer módulo es *WPcap.dll*, contiene funciones alto nivel como generación de filtros, buffering[19] de nivel de usuario además de funcionalidades avanzadas como estadísticas e inyección de paquetes.

En este tercer módulo los programadores pueden acceder a dos tipos de API:

- Las funciones contenidas en *packet.dll*, las cuales son mapeadas directamente a las llamadas en el nivel de kernel.
- El conjunto de funciones de más alto nivel provistas por *WPcap.dll* las cuales más amigables al usuario y más poderosas. Una llamada a una función en este módulo puede traducirse como un conjunto de funciones en el nivel NPF.

1.4.2. JNETPCAP

jNetPcap es un API de desarrollo de software para el (SDK[20]) de Java, cuya función básica es la proveer de un "*envoltorio java*"[21] para la biblioteca *libpcap*[22]. Este SDK también provee de funcionalidades adicionales no presentes en el proyecto nativo original libpcap.

El objetivo de jNetPcap es el de proveer una mayor facilidad para el desarrollo de aplicaciones típicas de la biblioteca libpcap.

jNetPcap consta del API de la biblioteca de clases contenido en un archivo .jar para el uso de una completa lista de operaciones de *libpcap*. El paquete es dependiente de la plataforma y por esa razón existe una biblioteca compartida nativa. En el caso de sistemas Win32 la biblioteca es un .dll y en sistemas Unix es

---

[19] Buffering es un término relativo al buffer de datos, en donde se almacenan datos para evitar que el programa o recurso que los requiere, ya sea hardware o software, se quede en algún momento sin datos.
[20] Software Development Kit
[21] El término original en inglés es *java wrapper*
[22] Biblioteca propia de sistemas Unix utilizada para captura de paquetes de red

un fichero .so. Esta biblioteca compartida provee de una interfaz "*JNI bridge*"[23] entre Java, la plataforma nativa de software y la biblioteca libpcap propiamente dicha.

Este paquete de desarrollo no contiene a libpcap y requiere de una instalación por separado, tanto para sistemas Unix y Win32. La versión de libpcap para Windows es WinPcap.

1.4.2.1. Estructura de jNetPcap

El kit de desarrollo de jNetPcap consta de dos partes:

- El libpcap wrapper, el cual provee de casi todas las funcionalidades de la biblioteca nativa de libpcap, pero para un ambiente java. Esta parte representa el nivel más bajo en el API del SDK de jNetPcap.
- El packet decoding framework, que consta de una colección de paquetes para la captura de paquetes y su posterior procesamiento y decodificación. Una vez realizado el procedimiento de decodificación, es posible acceder a mayor información dentro de estos paquetes usando un conjunto de clases java llamadas protocol headers o cabeceras de protocolo en su traducción al castellano.

Ambas partes pueden trabajar de manera independiente, es decir, que no necesitan realizar llamadas al API del otro para realizar métodos de sus clases. Sin embargo, pueden trabajar en conjunto sin ninguna dificultad, por ejemplo el libpcap wrapper puede entregar los paquetes al packet decoding framework para su procesamiento, aunque este framework posea de métodos para el escaneo y decodificación de cualquier paquete almacenado en memoria.

A continuación se muestra un esquema de la estructura de los paquetes de clases en el SDK de jNetPcap y una breve descripción de los mismos[24]:

---

[23] Java Native Interface (JNI) es un framework de programación que permite que un programa escrito en Java pueda interactuar con programas escritos en otros lenguajes.
[24] Referencia: http://www.jnetpcap.org/node/135

org
+-> jnetpcap – este es el paquete principal del libpcap wrapper. Este contiene
+ todo el API para el acceso a las funcionalidades de libpcap.
| Estas clases y métodos son pequeños y simplemente envían las
| peticiones (requests) sobre la librería libpcap nativa, el cual se encarga
| de realizar las acciones requeridas.
|
+-> winpcap – es una extensión para el libpcap wrapper que provee las
| funciones WinPcap. Este paquete es dependiente del sistema operativo
| y se debe ejecutar la llamada WinPcap.isSupported() antes de usar
| cualquiera de las clases o métodos de este paquete.
|
+-> nio – son las clases pertenecientes al paquete nativo de E/S y memoria.
| Este paquete define las clases de administración de memoria que son
| asignadas a la memoria nativa, es decir, que las funciones y estructuras
| nativas tienen su par en clases de java. Aquí se encuentra la clase
| JBuffer, una muy importante.
|
+-> util – varias clases de utilidades. En este paquete se pueden encontrar
| clases para administración de configuraciones de jNetPcap, resolución
| de nombres, entre otros.
|
+-> packet - packet decoding framework. Este paquete define importantes
| componentes del decodificador. JScanner, PcapPacket, y las
| importantes clase base JHeader. JScanner decodifica los paquetes
| y almacena la información del estado del paquete en estructuras
| nativas. La clase PcapPacket lee esta información de estado y puede
| emparejar objetos header (referencia a memoria nativa).
| Todas las clases header (cabecera) son subclase de JHeader.
|
+-> header – es una biblioteca de los CORE protocol headers soportados
| Núcleo de Cabeceras de Protocolo). Aquí se encuentran las
| definiciones los headers Ip4, Tcp, Udp, Ethernet, entre otros, listos y
| accesibles para su utilización.
|
+-> format – formateo de paquetes. Estos arrojan el contenido del paquete
| decodificado en forma textual. Se puede hacer uso del TextFormatter
| o el XmlFormatter.
|
+-> structure – este paquete es utilizado por los protocol builder
| (constructores de protocolos).
|
+-> annotate – son interfaces utilizadas en la escritura de protocol headers
(cabeceras de protocolo). Sin embargo en la escritura de la
definición de un nuevo protocol header no se necesita acceder a
ninguna de estas, simplemente está allí para marcar métodos con
una connotación especial y proveer de algunos parámetros
exactos.

1.5. PLATAFORMA DE DESARROLLO DE SOFTWARE

Una plataforma de desarrollo constituye el entorno de software comúnmente ligado a un sistema operativo, a un lenguaje de programación o a un API. Sobre la cual es posible llevar a cabo todos los procesos necesarios para la creación de una aplicación.

1.5.1. JAVA [25]

Java es un lenguaje de programación orientado a objetos desarrollado por Sun Microsystems. Este lenguaje contiene sintaxis de c y c++, pero su modelo de objetos es muy simple, ya que elimina herramientas de bajo nivel, que generalmente causa la mayoría de errores, como la manipulación directa de punteros o memoria.

Este lenguaje es muy famoso ya que es independiente de cualquier plataforma, además como fue basado en el lenguaje c y c++ es muy flexible para su aprendizaje y manipulación.

Java se ha construido con extensas capacidades de conexión como TCP/IP, por lo tanto, contiene librerías de rutinas para acceder e interactuar con protocolos como ftp y http. Esto permite al programador acceder información a través de la red con tanta facilidad como ficheros locales.

Java posee las características de ser orientado a objetos, robusto, seguro, portable, multithreaded[26], dinámico, a pesar de no ser distribuido permite que sus aplicaciones lo puedan ser, y por último posee una arquitectura neutral.

Java además de ser un lenguaje de programación, proporciona herramientas de desarrollo para la creación de programas en Java llamado JDK (**Java Development Kit**).

---

[25] Referencia: http://java.sun.com/javase/downloads/index.jsp (javadoc JDK 6)
[26] Significa que varios threads se ejecutan simultáneamente en un espacio de direcciones compartido.

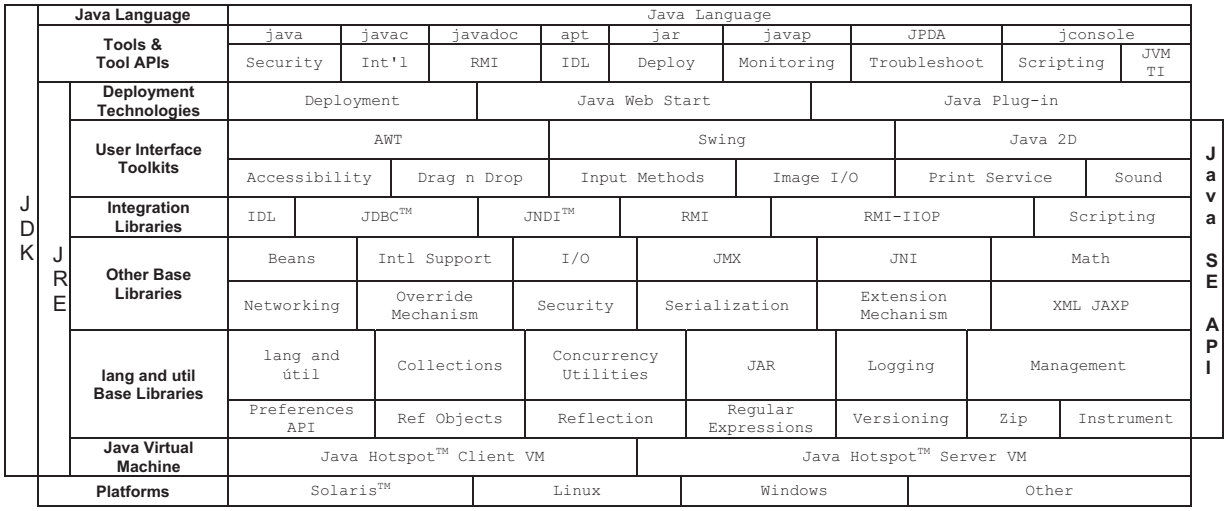


Gráfico 1.23 Arquitectura de Java

1.5.1.1. JAVARUNTIME ENVIROMENT (JRE) [27]

Es un conjunto de utilidades que permite la ejecución de programas java. Incluye todos los módulos de ejecución necesarios, incluyendo la Máquina Virtual Java (JVM)[28], para ejecutar aplicaciones hechas con Java.

Además contiene las clases principales de Java (clases core), y ficheros de soporte. JRE es el componente de ejecución del Java Developer Kit (JDK) de Sun. También incluye el Java Plug-in.

El Java Plug-in posee la posibilidad de usar la ultimísima implementación Java de Sun en los navegadores Internet Explorer, Netscape Navigator, Mozilla y Firefox, en lugar de usar la máquina virtual de Java (JVM) que el navegador trae por defecto. Incluye un soporte total del JDK: permitiendo a los desarrolladores de empresas hacer applets[29] Java, usando todas las funciones del JDK (ejemplo: RMI[30], JavaBeansTM, etc.).

---

[27] Referencia: http://www.java.com/es/about/

[28] (JVM)Programa ejecutable en una plataforma específica, capaz de interpretar e ejecutar instrucciones expresadas en código binario, generadas por un compilador de lenguaje java.

[29] Componente de una aplicación que se ejecuta en el contexto de otro programa (navegador web).

[30] (RMI) es un mecanismo ofrecido por Java para invocar un método de manera remota.

1.5.1.2. NETBEANS [31]

Es una plataforma en la cual se puede desarrollar aplicaciones a partir de un conjunto de componentes de software llamados módulos. Un módulo es un archivo Java que contiene clases de java, que están escritas para interactuar con las APIs de NetBeans y un archivo especial (manifest file) que lo identifica como módulo.

Además es una herramienta de programación pensada para escribir, compilar, depurar y ejecutar programas.

Existe además un número importante de módulos para extender el IDE [32] NetBeans. Por otro lado es un producto libre y gratuito sin restricciones de uso, y fue escrito en Java, pero puede servir para cualquier otro lenguaje de programación.

Entre las principales características de NetBeans es de ser una base modular y extensible usada como una estructura de integración, que sirve para crear aplicaciones de escritorio grandes.

Las empresas independientes asociadas, especializadas en desarrollo de software, proporcionan extensiones adicionales que se integran fácilmente en la plataforma y que pueden también utilizarse para desarrollar sus propias herramientas y soluciones, permitiendo así abarcar más campos para el desarrollo de aplicaciones.

La plataforma ofrece servicios comunes a las aplicaciones de escritorio, permitiéndole al desarrollador enfocarse en la lógica específica de su aplicación.

---

[31] Referencia: http://www.netbeans.org/
[32] (IDE) entorno de desarrollo integrado.

Entre las características de la plataforma están:

- Administración de las interfaces de usuario (menús y barras de herramientas).
- Administración de las configuraciones del usuario
- Administración del almacenamiento (guardando y cargando cualquier tipo de dato).
- Administración de ventanas
- Framework basado en asistentes (diálogo paso a paso).

1.5.1.3. JSC 1.0 [33]

JSC se lo puede considerar más que una biblioteca de componentes, ya que esta permite proporcionar la herramienta para el desarrollo de aplicaciones estadísticas como: plataforma de código abierto independiente de los paquetes estadísticos generales, paquetes especializados para análisis no tradicionales como Bayesiano[34], muestreo, estadística multivariante, gráficas, simulaciones, animaciones, etc.

JSC es una biblioteca de componentes reutilizables y extensibles para la creación de software estadístico. Las bibliotecas estándar de Java ofrecen muchas clases que podrían ser útiles en aplicaciones estadísticas. Estas cubren áreas como funciones matemáticas, estructura de datos tablas, manejo de archivos, gráficos, y componentes de interfaz. Sin embargo algunas de estas bibliotecas relacionadas con gráficos y componentes de interfaz son muy complejos para principiantes. Por ello JSC contiene algoritmos que van más allá de funciones básicas matemáticas, y de generación de números aleatorios.

Pocos algoritmos estadísticos y numéricos se han publicado en Java, la mayoría de estos han publicado en FORTRAN, un lenguaje difícil de traducir a este lenguaje.

JSC ha permitido proporcionar una herramienta de desarrollo de software para aplicaciones estadísticas más avanzadas específicamente para Java.

---

[33] Referencia: http://www.jsc.nildram.co.uk/

[34] Un tipo de probabilidades subjetivas usadas como filtros para correo basura (spam).

Esta biblioteca proporcionará:

- Todos los gráficos de alto nivel que se puede esperar, como histogramas, diagramas de dispersión y boxplots[35] y gráfica de bajo nivel que le permiten construir fácilmente su propia muestra gráfica representando en un sistema de coordenadas naturales.
- Versiones simplificadas de los componentes de interfaz de Java, tales como menús, barras de desplazamiento, cajas de diálogo, y componentes de alto nivel, tales como ventanas de datos similares a los encontrados en los paquetes estadísticos.
- Las funciones y operaciones básicas útiles para la estadística tales como la ordenación, ranking, funciones para distribuciones de probabilidad y procedimientos para la evaluación y diferenciación de funciones matemáticas de entrada ingresadas por el usuario.
- Algoritmos estadísticos que abarcan muchos aspectos de las estadísticas: en particular, descriptivo, tradicional, las estadísticas no paramétricas; ajuste de curvas y regresión; distribución y la generación de números aleatorios.

De esta manera JSC se convierte en una biblioteca pilar para los desarrolladores de software con aplicaciones estadísticas.

1.5.2. MYSQL[36]

MySQL es un sistema de gestión de base de datos relacional, que proporciona un servidor de base de datos SQL (Structured Query Language).

Entre sus características más destacables tenemos: rapidez, multiusuario, multi-threaded y multi-procesador, especialmente diseñado para entornos de producción críticos y alta carga de trabajo.

---

[35] Es un gráfico basado en cuartiles llamado comúnmente diagrama de caja.
[36] Referencia: http://dev.mysql.com/, http://dev.mysql.com/doc/refman/5.0/es/introduction.html

El servidor puede ser utilizado en sus dos modalidades: autónomo para aplicaciones cliente/servidor en un entorno de red y en forma de biblioteca para ser embebido en aplicaciones individuales o donde no existe una red disponible.

En cuanto a seguridad posee un sistema de privilegios y contraseñas muy seguro con verificación basada en el host. Usa encriptación para el transporte de contraseñas para asegurar la conexión con el servidor.

MySQL soporta a grandes bases de datos que pueden contener 50 millones de registros. Existen usuarios que usan MySQL Server con 60.000 tablas y cerca de 5.000.000.000.000 de registros.

Los clientes pueden conectar con el servidor MySQL usando sockets TCP/IP en cualquier plataforma. También existe soporte para la implementación de programas cliente que usen ODBC[37] tanto en Windows como Unix y JDBC[38] para los desarrolladores de aplicaciones en Java.

MySQL es una solución real para grandes cargas transaccionales y de datos.

Para mayor información se recomienda leer la documentación existente en su página web.

1.5.3. JFREECHART[39]

JFreeChart es una biblioteca gratuita y de código abierto para la generación de gráficos para la plataforma Java. Fue diseñada para su uso en aplicaciones de escritorio, applets[40], servlets[41] y JSP[42].

Esta biblioteca puede generar gráficas de pastel, diagramas de barras (regulares, apiladas y con un efecto 3D opcional), diagramas de líneas, diagramas de dispersión, diagramas de series de tiempo, diagramas de Gantt, diagramas de

---

[37] Open Database Connectivity
[38] Java Database Connectivity
[39] Referencia: http://www.jfree.org/jfreechart/
[40] Es un componente o subprograma que ofrece información gráfica, interacción con el usuario y tiene privilegios de seguridad restringidos que se ejecuta por otro programa como por ejemplo un navegador web.
[41] Es un programa que se ejecuta en un servidor principalmente para la generación de páginas web dinámicas a partir de las peticiones enviadas desde un navegador web.
[42] Es una tecnología de Java similar a un servlet en cuanto a resultados finales pero de manera más simplificada.

magnitud (dial, compás y termómetro), diagramas de símbolos, diagramas combinados y más.

También ofrece otras características como zoom interactivo, exportación de imágenes en formato PNG y JPG, manejo de eventos de mouse, información del gráfico mediante texto emergente (tooltips), generación de mapas de imágenes HTML entre los principales.

1.5.4. NATIVE SWING[43]

La biblioteca NativeSwing permite una fácil integración de algunos componentes del sistema operativo nativo dentro de aplicaciones Swing[44] de Java, además de algunas utilidades nativas para mejorar el API de Swing. Está compuesta de una biblioteca que funciona como framework y de muchos componentes desarrollados en base a SWT[45].

Entre las implementaciones más importantes de componentes SWT están el navegador web, el reproductor de animaciones flash, el reproductor multimedia, un editor HTML y un resaltador de sintaxis.

1.5.5. JCALENDAR [46]

La biblioteca JCalendar implementa un selector gráfico de fecha. JCalendar está compuesto de varias clases individuales como el JDayChooser, el JMonthChooser y el JYearChooser. Cada una de estas incluye la propiedad locale[47] y un set iconos para ser usados en aplicaciones gráficas.

Como parte del paquete de distribución también existe un JDateChooser, una clase que consta de un JDateEditor (para la edición directa de la fecha) y un botón para abrir una instancia de JCalendar para seleccionar una fecha.

---

[43] Referencia: http://djproject.sourceforge.net/ns/

[44] Biblioteca gráfica de Java con elementos para la implementación de la interfaz gráfica de usuario.

[45] Standard Widget Toolkit, es un conjunto de componentes desarrollados por Eclipse para construir interfaces gráficas en Java, utiliza componentes nativos para hacerla consistente con el sistema operativo actual.

[46] Referencia: http://www.toedter.com/

[47] En Java un objeto Locale representa una región geográfica específica, política o cultural.

CAPÍTULO 2

2. REQUERIMIENTOS

La redacción de un documento de requerimientos de software debe seguir cierta formalidad pues es la declaración oficial del software que se implementará. El estándar más ampliamente divulgado es el IEEE/ANSI 830-1998. En el presente proyecto de titulación el documento de requerimientos se basa en la estructura sugerida por este estándar.[1]

El estándar IEEE/ANSI 830-1998 sugiere la tabla 2.1 que describe los requerimientos contenidos en 3 secciones.

Table of Contents

1. Introduction
   - 1.1 Purpose
   - 1.2 Scope
   - 1.3 Definitions, acronyms, and abbreviations
   - 1.4 References
   - 1.5 Overview
2. Overall description
   - 2.1 Product perspective
   - 2.2 Product functions
   - 2.3 User characteristics
   - 2.4 Constraints
   - 2.5 Assumptions and dependencies
3. Specific requirements (See 5.3.1 through 5.3.8 for explanations of possible specific requirements. See also Annex A for several different ways of organizing this section of the SRS.)

Appendixes

Index

Tabla 2.1 Especificaciones de Requerimientos de Software (ERS).[2]

Como se muestra en la tabla 2.1 sección 1 sugiere una introducción que contiene propósito, alcance y referencias. Estas mismas están descritas en la propuesta del proyecto de titulación mostrada en el anexo F. En la sección 2 sugiere una descripción general del producto, así como en la sección 3 incluye los requerimientos específicos del software, los cuales se detallarán a continuación.

---

[1] SOMMERVILLE, Ian; Ingeniería del Software; Editorial Pearson; Séptima Edición

[2] Anexo F - PLAN DEL PROYECTO DE TITULACIÓN.

2.1. Descripción general

2.1.1. Perspectiva del Producto

En la actualidad el compartir recursos informáticos virtualmente ha sido un pilar fundamental para el desarrollo de las diferentes tecnologías. El Internet siendo uno de los principales caminos para el intercambio de información, se ha convertido en un recurso muy valioso que se lo debe administrar cuidadosamente, por ello ha existido la necesidad de crear herramientas que permitan administrar de mejor manera este recurso.

El presente proyecto está dirigido a personas con un nivel intermedio de conocimiento en redes informáticas, con el fin de que aprovechen todas las funcionalidades del software a desarrollar.

La mayoría de herramientas de monitorización de ancho de banda, muestran limitaciones en cuanto a su capacidad de tratamiento de datos. Por ello surge la necesidad de proveer una herramienta capaz de profundizar el análisis de estos datos. En base a los resultados obtenidos del análisis el usuario podrá tomar decisiones que aporten al uso óptimo del servicio de Internet.

2.1.2. Funciones del producto

Los requerimientos iniciales se obtuvieron en base a los objetivos generales y específicos del proyecto, además se tuvieron en cuenta aportes tanto de requerimientos técnicos y necesidades que surgieron al administrar y evaluar una red por parte de los desarrolladores de este proyecto de titulación.

Este software está orientado a un usuario con perfil de administrador de red, que desea obtener información sobre el comportamiento de una red informática que se conecte a Internet en tiempo real y que permita de manera gráfica representar los datos obtenidos durante el uso de este recurso.

Además permite la obtención de parámetros estadísticos, que ayudarán al administrador de red a interpretar el comportamiento de la red a través de gráficos que representen estas variaciones.

El software mostrará resultados por cada computador y sus diferentes conexiones a Internet, manteniendo un sistema de datos que respalde la información que se ha captado durante el uso de Internet, de esta manera el administrador tendrá la libertad de evaluar intuitivamente cómo se comporta la red comparando los datos anteriores con los nuevos datos obtenidos con este software.

Se incluirán recomendaciones que guíen al administrador de red sobre qué acción tomar frente a situaciones que comprometan el rendimiento y disponibilidad de la conexión a Internet.

Este software permitirá interactuar al usuario con una interfaz gráfica amigable, y permitir el acceso a todas las diferentes herramientas mencionadas anteriormente.

2.1.3. Características del usuario

El presente proyecto está orientado para un usuario que tenga conocimientos en administración de redes informáticas, nociones básicas de estadística y redes TCP/IP, con el objetivo de aprovechar todas las herramientas del software para la optimización del consumo de Internet.

2.1.4. Restricciones generales

La aplicación final debe ser capaz de ejecutarse en ambiente Windows a partir de XP, y por lo menos en dos distribuciones de Linux.

La captura de paquetes debe enfocarse exclusivamente al tráfico de Internet.

Los costos de desarrollo e implementación del software deben ser los mínimos posibles.

2.1.5. Suposiciones

Se supone que el tráfico a escucharse pertenece a la intranet en la cual el software está realizando el análisis de tráfico, mas no puede realizar el monitoreo de otras subredes que no estén interconectadas físicamente a la interfaz de escucha utilizado por el programa dentro del equipo activo para la salida a

Internet, a no ser que se configure el equipo activo para que se replique el tráfico de dichas subredes en la interfaz de red que se está monitoreando.

2.2. Requerimientos específicos

2.2.1. Especificación de requerimientos del sistema

- El software permitirá realizar monitorización en tiempo real del tráfico de red. Esto implica la entrega de gráficos de la velocidad de transmisión utilizado a través del tiempo.

  Debe existir soporte para dos tipos de gráficos básicos:

  - ☐ La velocidad de transmisión para el tráfico entrante / tráfico saliente (incoming traffic / outgoing traffic) versus la variable tiempo, para el tráfico total de la red.

  - ☐ La velocidad de transmisión para el tráfico entrante / tráfico saliente (incoming traffic / outgoing traffic) versus la variable tiempo, para todos los protocolos o un protocolo específico, eligiendo las estaciones de trabajo a monitorear.

- En general el software debe permitir diferenciación de tráfico, es decir, que sea posible determinar el tráfico correspondiente a un host, puertos o protocolos de capa aplicación. El usuario deberá tener acceso a esta información en tiempo real mediante la implementación de un sniffer para la visualización de las cabeceras de los protocolos. Además esta información deberá ser almacenada de forma que pueda ser utilizada en el futuro en caso de ser necesaria.

- Utilizando los datos almacenados el software deberá proveer resúmenes gráficos o tablas sobre el tráfico de Internet para las distintas estaciones de trabajo que están siendo monitoreadas, tanto para el tráfico entrante (incoming traffic) y el tráfico saliente (outgoing traffic), priorizando una fácil comprensión de los mismos. El usuario elegirá el rango de tiempo desde el cual se realizarán los cálculos.

El software debe generar los siguientes resúmenes gráficos:

- Tasa de transferencia promedio del uso de Internet.

  Permitirá observar el promedio de tasa de transferencia de datos de entrada o salida mediante un diagrama de barras y se podrá diferenciar estaciones de trabajo y protocolos en general, para el rango de tiempo seleccionado.

- Porcentajes de uso de tráfico de Internet.

  Mediante un diagrama de pastel se observarán los porcentajes de uso de tráfico de Internet para las estaciones de trabajo seleccionadas, diferenciando protocolos y puertos en caso de tráfico TCP/UDP.

- Reconstrucción del historial de la base de datos usando diagramas de líneas.

  Por medio de un diagrama de líneas se recuperarán los valores de la base de datos de tráfico de Internet para las estaciones de trabajo, protocolos y rango de tiempo seleccionados.

- Reconstrucción del historial de la base de datos usando diagramas de pasos.

  Por medio de un diagrama de pasos se graficará el flujo de datos de tráfico de Internet para las estaciones de trabajo, protocolos y rango de tiempo seleccionados.

- DNS reverso y ranking de las direcciones IP con mayor flujo de entrada de datos.

  Mediante un diagrama de barras deberá mostrarse la cantidad de bits transferidos para las distintas direcciones IP registradas en la base de datos por cada host seleccionado. Debe incluirse un mecanismo que realice el proceso de DNS reverso para la resolución de nombres de las direcciones IP.

- Utilizando los datos almacenados el software deberá proveer de parámetros estadísticos básicos, con cálculos basados en Estadística Descriptiva tanto para el tráfico entrante (incoming traffic) y el tráfico saliente (outgoing traffic). El usuario elegirá el rango de tiempo desde el cual se realizarán los cálculos.

  El software debe generar el siguiente resumen estadístico:

  - ☐ Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet.

    Deberá graficar un histograma de frecuencias para los valores obtenidos de la base de datos, para las estaciones de trabajo y protocolos seleccionados, estableciendo rangos uniformes para los valores de ancho de banda. Conjuntamente debe graficarse la distribución de frecuencias acumuladas.

- Para un análisis individual más profundo de cada protocolo, este software utilizará los datos almacenados para la obtención de series de tiempo. El usuario deberá establecer el rango de tiempo desde el cual se extraerán los datos.

  El software deberá graficar series de tiempo para los distintos protocolos y estaciones de trabajo, eligiendo el intervalo regular de tiempo más adecuado de manera automática, de acuerdo al rango elegido por el usuario.

  El software debe generar los siguientes gráficos de series de tiempo:

  - ☐ Series de Tiempo usando diagramas de líneas.

  - ☐ Series de Tiempo usando diagramas de pasos.

- El software debe proveer mecanismos para guardar los gráficos y texto generados a partir de los datos obtenidos de la captura de tráfico de Internet.

2.2.2. Requerimientos del Producto

En los diferentes requerimientos del presente software se menciona:

- Portabilidad.- El software deberá ejecutarse sobre la máquina virtual de java (JVM), esto permitirá que el programa a desarrollarse se ejecute en Windows(Windows XP SP2, Windows 2003 server, Windows Vista, Windows 7) y Linux (CentOS 5.2, Ubuntu 8.02), haciendo uso de bibliotecas especializadas, proporcionando mayor robustez y eficiencia.
- Robustez.- El programa deberá ser robusto en cuanto a captura de paquetes y almacenamiento de datos, ya que para el análisis de los mismos se requiere que las diferentes muestras sean fiables, independientemente si fueron capturadas en tiempo real o almacenadas en una base de datos.
- Fiabilidad.- Los datos almacenados deberán estar disponibles en cualquier momento que el usuario requiera, para el tratamiento de los mismos. En lo posible el programa no deberá tener ningún tipo de fallos, ya que la inoperatividad del mismo introducirá errores al momento de evaluar los resultados del análisis del uso del servicio de Internet.
- Facilidad de uso.- Para el usuario final con las características descritas anteriormente, las herramientas a desarrollarse deberán ser intuitivas para su manejo, además el software deberá incluir recomendaciones que orienten de mejor manera en la toma de decisiones. Debe poseer un tutorial de video y animaciones que expliquen el funcionamiento del mismo.
- Recursos de hardware.- El software deberá en lo posible utilizar los mínimos recursos del sistema, tratando de no alterar el rendimiento total del mismo. Debe ser capaz de ejecutarse sin ningún inconveniente en una PC Pentium 4 1.6 GHz/ AMD Atlon 1.2 GHz y 1GB de RAM.
- Rapidez.- El software a desarrollar deberá mostrar los diferentes datos y peticiones realizadas por el usuario en forma concisa. Se debe buscar la forma de tener el menor retardo posible en la actualización de pantallas y procesos internos realizados durante los diferentes cálculos matemáticos.

CAPÍTULO 3

3. DESARROLLO DEL SOFTWARE

Una vez alcanzada esta etapa es oportuno preguntar ¿Cuál es el lenguaje de programación elegido para el desarrollo de la solución?, ¿Qué metodología de diseño se utilizará?.

En respuesta a la primera pregunta, se optó por un lenguaje de programación orientado a objetos que cumpla con el requisito fundamental de la gratuidad para disminuir el costo del proyecto, otros factores importantes que se tomaron en cuenta fueron los conocimientos previos, que sea capaz de entenderse de manera intuitiva, la familiaridad con el lenguaje, la disponibilidad bibliotecas especializadas y herramientas de desarrollo.

Para la segunda interrogante es obvio que la metodología de diseño debe ser compatible con el lenguaje de programación, suficientemente descriptiva, ampliamente aceptada y con bibliografía disponible para consulta.

De acuerdo a estos antecedentes y requerimientos de portabilidad para el presente proyecto de titulación se eligió a Java como el lenguaje apropiado para el desarrollo del software y como metodología de diseño a UML.

3.1. DIAGRAMA DE CASOS DE USO

De acuerdo a los requerimientos del usuario, se describe el comportamiento del sistema en un diagrama fácil de comprender, incluso de forma intuitiva, como es el diagrama de casos de uso mostrado a continuación.

El desarrollo de cada caso de uso se detallará en la sección 3.7.

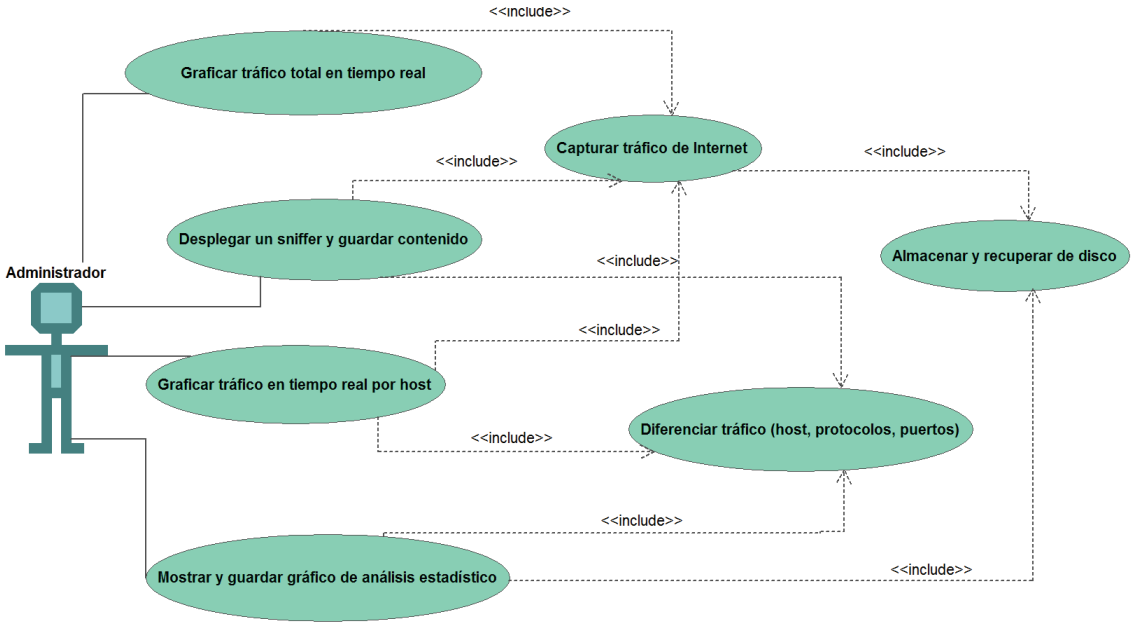


Gráfico 3.1 Diagrama de Casos de Uso para los requerimientos del usuario

3.2. DESCRIPCIÓN GENERAL DEL SISTEMA

Luego de un análisis exhaustivo de los requerimientos, se ha planteado la solución del problema como dos subproyectos diferenciados por su funcionalidad principal:

- TrafficStatistics: subproyecto gráfico exclusivo de todos los casos de uso que requieren captura de tráfico de Internet.
- QueryStatistics: subproyecto gráfico dedicado al análisis estadístico y resúmenes informativos de los datos almacenados por el subproyecto anterior.

3.3. BASE DE DATOS COMO SOLUCIÓN PARA ALMACENAMIENTO DE INFORMACIÓN DE TRÁFICO DE INTERNET EN DISCO

El presente proyecto de titulación tiene como principal objetivo la generación de resúmenes estadísticos. Por naturaleza este tipo análisis requieren realizarse sobre datos recopilados a lo largo del tiempo, por lo tanto es necesario almacenarlos de forma permanente para acceder a ellos en cualquier momento.

Los requerimientos incluyen la *diferenciación de tráfico* como algo fundamental en este proyecto, es decir, recuperar información de acuerdo a parámetros como estaciones de trabajo, protocolos y puertos.

Una solución que reúna las condiciones antes expuestas, es sin lugar a dudas una base de datos. Uno de los objetivos del proyecto de titulación es hacer que el uso de la aplicación sea lo más simple posible, en consecuencia se convierte en una necesidad embeber la base de datos en la aplicación y que su funcionamiento sea transparente para el usuario.

El gestor de base de datos elegido por poseer estos atributos es MySQL, específicamente MySQL Connector/MXJ 5.0 el cual es un paquete Java para desarrollo y administración de una base de datos, que puede ser embebida dentro una aplicación Java existente. Para la inicialización de la base de datos se emplea parámetros en el url de conexión JDBC, esto implica que la base de datos se inicializará al realizarse la primera conexión sin que el usuario final tenga complicaciones de instalación.

MySQL Connector/MXJ hace que la base de datos parezca ser un objeto basado en Java. El procedimiento que el conector realiza para que esto sea posible es el siguiente: determinando la plataforma nativa, seleccionando el archivo binario apropiado y lanzando el ejecutable.

3.3.1. DISEÑO DE LA BASE DE DATOS

Siguiendo el diseño de una base de datos relacional, tomando en cuenta todos los valores y variables necesarios para identificar y diferenciar el tráfico de Internet para cada estación de trabajo, se llegó al siguiente modelo mostrado en el siguiente gráfico.

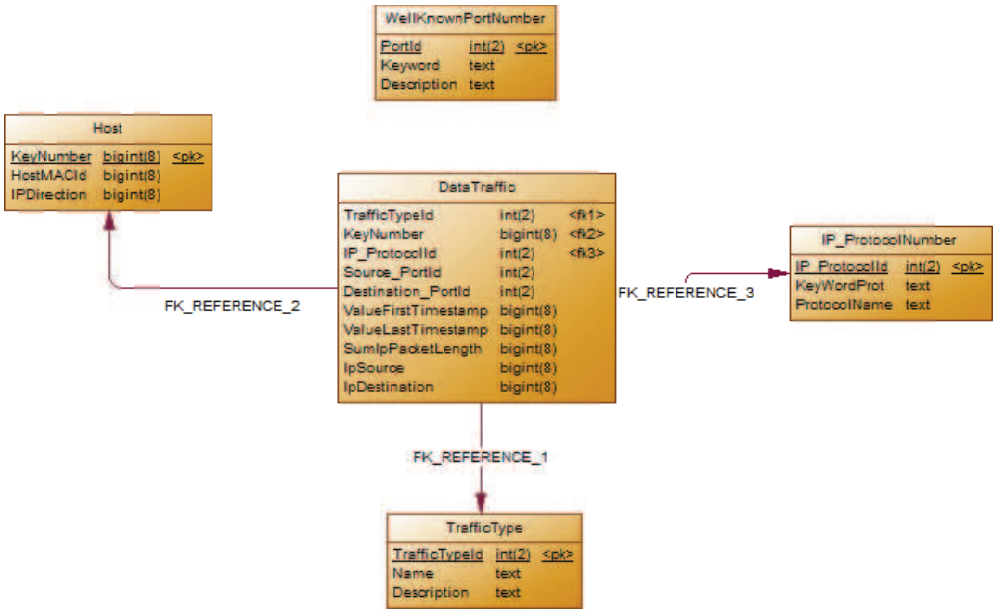


Gráfico 3.2 Modelo relacional de la base de datos

3.3.1.1. Descripción del modelo

Tabla Host

Asigna un identificador entero autoincremental a cada estación de trabajo que va siendo ingresada en la tabla. El host es diferenciado por su MAC y dirección IP almacenados en un valor entero equivalente a su representación en octetos.

Tabla TrafficType

Identifica los tipos de tráfico existentes en una conexión a Internet, el tráfico entrante y el tráfico saliente. El tráfico entrante es identificado con el entero 0 y el saliente con el 1. Los otros campos de la tabla describen brevemente cada tipo de tráfico.

Tabla IP_ProtocolNumber

Lista cada uno de todos los posibles valores del campo protocolo dentro la cabecera del paquete IP que identifican los protocolos del siguiente nivel, asignados de acuerdo al RFC 1700, se incluyen los campos de sus siglas y nombre completo.

Tabla WellKnownPortNumber

Lista todos los puertos en el rango de 1 a 1024 denominados puertos bien conocidos, con sus siglas y descripción de acuerdo al RFC 1700. Estos valores son usados por los protocolos de transporte TCP y UDP para identificar una conexión.

Tabla DataTraffic

Tabla principal donde se añadirán los registros con la información recopilada en la captura de tráfico de Internet. Cada registro indica una conexión de una estación de trabajo de la red monitoreada con el exterior. Dada la importancia de esta tabla se describirán cada uno de sus campos:

TRAFFICTYPEID: clave externa para identificar el tipo de tráfico (entrante o saliente).

KEYNUMBER: clave externa que identifica la estación de trabajo a la que pertenece el registro.

IP_PROTOCOLID: clave externa para identificar el protocolo de capa superior en la cabecera del paquete IP.

SOURCE_PORTID: entero que indica el puerto origen en caso de que este exista. Este campo puede ser nulo en comunicaciones donde no se utilicen protocolos de capa transporte como ICMP.

DESTINATION_PORTID: entero que indica el puerto destino en caso de que este exista. Este campo puede ser nulo en comunicaciones donde no se utilicen protocolos de capa transporte como ICMP.

VALUEFIRSTTIMESTAMP: entero que almacena el valor del timestamp del primer paquete capturado de esta conexión.

VALUELASTTIMESTAMP: entero que almacena el valor del timestamp del momento inmediatamente anterior al ingreso del registro.

SUMIPPACKETLENGTH: entero que indica la cantidad total en bytes de los paquetes IP capturados incluidos las cabeceras, para esta conexión, en el lapso de tiempo comprendido desde el primer timestamp hasta el último.

IPSOURCE: dirección IP origen representada en su valor entero correspondiente. Puede ser privada o pública si el tráfico es saliente o entrante respectivamente.

IPDESTINATION: dirección IP destino representada en su valor entero correspondiente. Puede ser pública o privada si el tráfico es saliente o entrante respectivamente.

Nota: El valor del timestamp corresponde a la diferencia, medida en milisegundos, entre la hora actual y la medianoche del 1ero de Enero de 1970.

3.3.1.2. Procedimientos almacenados

Los procedimientos almacenados permiten realizar rutinas de actualización y consultas a los registros de la base de datos. Esto se vuelve útil principalmente cuando se lo hace con mucha frecuencia y se quiere evitar el tener que escribir todo el conjunto de procedimientos necesarios, para llevar a cabo una operación sobre la base de datos.

Dos operaciones son bastante repetitivas en el presente caso:

- Añadir un nuevo host.

```
create procedure SP_INSERTHOST (IN HOSTMACID_IN bigint(8), IN IPDIRECTION_IN bigint(8))
BEGIN
INSERT INTO HOST ( HOSTMACID, IPDIRECTION) VALUES(HOSTMACID_IN, IPDIRECTION_IN);
END;
```

- Añadir un nuevo registro de tráfico.

```
create procedure SP_DATATRAFFIC (IN TRAFFICTYPEID_IN int(2), IN KEYNUMBER_IN bigint(8), IN IP_PROTOCOLID_IN int(2),IN SOURCE_PORTID_IN int(2),IN DESTINATION_PORTID_IN int(2), IN VALUEFIRSTTIMESTAMP_IN bigint(8), IN VALUELASTTIMESTAMP_IN bigint(8),IN SUMIPPACKETLENGTH_IN bigint(8), IN IPSOURCE_IN bigint(8), IN IPDESTINATION_IN bigint(8))
BEGIN
INSERT INTO DATATRAFFIC ( TRAFFICTYPEID, KEYNUMBER, IP_PROTOCOLID, SOURCE_PORTID, DESTINATION_PORTID,VALUEFIRSTTIMESTAMP,VALUELASTTIMESTAMP, SUMIPPACKETLENGTH, IPSOURCE, IPDESTINATION) VALUES( TRAFFICTYPEID_IN, KEYNUMBER_IN, IP_PROTOCOLID_IN, SOURCE_PORTID_IN, DESTINATION_PORTID_IN, VALUEFIRSTTIMESTAMP_IN,VALUELASTTIMESTAMP_IN, SUMIPPACKETLENGTH_IN, IPSOURCE_IN, IPDESTINATION_IN );
END;
```

3.4. DESCRIPCIÓN DE LA BIBLIOTECA PRINCIPAL DE CLASES

La programación orientada a objetos permite desarrollar sistemas de software de manera modular, basada en los conceptos de clases y objetos. Bajo esta premisa se diseñó y desarrolló un conjunto de clases organizadas en paquetes, de acuerdo a su funcionalidad y objetivo, para ser usadas luego como una biblioteca en las aplicaciones gráficas de interfaz de usuario.

3.4.1. PAQUETES DE CLASES Y SU RELACIÓN CON EL DIAGRAMA DE CASOS DE USO

En la etapa de diseño se buscó cumplir con todos los requerimientos incluidos como casos uso en el gráfico 3.1 del capítulo.

Cada caso de uso a su vez puede ser desglosado como un gran conjunto de secciones de menor tamaño, que trabajando conjuntamente cumplen con el objetivo propuesto.

Tomando esto como base, a continuación se listan los paquetes de clases afines a cada caso de uso:

Casos de Uso	Paquetes de Clases
-Almacenar y recuperar de disco	-application.database -application.databaseScripts -application.dataquery -application.importexportdatabase
-Capturar tráfico de Internet	-application.datacapture -application.dinamicstructures
-Desplegar un sniffer y guardar contenido -Graficar tráfico en tiempo real por host -Diferenciar tráfico (host, protocolos, puertos)	-application.sniffer -application.realtimegraphs -application.save -application.newhostdetected
-Graficar tráfico total en tiempo real	-application.realtimegraphs
-Mostrar y guardar gráfico de análisis estadístico -Diferenciar tráfico (host, protocolos, puertos)	-application.graphicsDialogs -application.statisticalgraphics -application.save -application.descriptive -application.reversednsresolver
*Complementos de ayuda y mejoramiento visual (*) Mejoras adicionales que no forman parte de los casos de uso.	-application.lookandfeel -application.flashVideo -application.flashVideo.resources -application.minibrowser -application.sugerencia -application.urlsSugerencias

Tabla 3.1 Relación de casos de uso con paquetes de clases

3.4.2. DESCRIPCIÓN DE LOS PAQUETES DE CLASES

La biblioteca contiene un paquete principal denominado *application* que contiene todo el árbol de paquetes desarrollados, como se muestra a continuación:

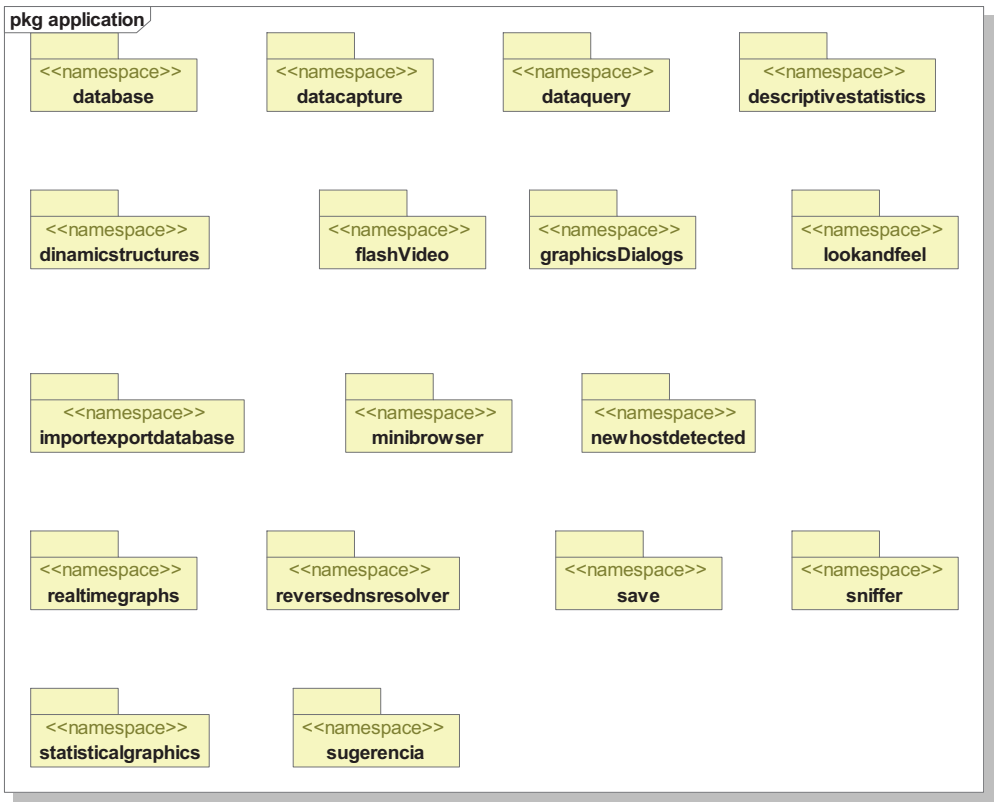


Gráfico 3.3 Paquetes de clases

3.4.2.1. Paquete database

Contiene clases dedicadas para la creación, establecimiento, consultas, inserción de nuevos registros de tráfico de Internet, visualización en consola de las consultas realizadas y cierre de la conexión de la base de datos.

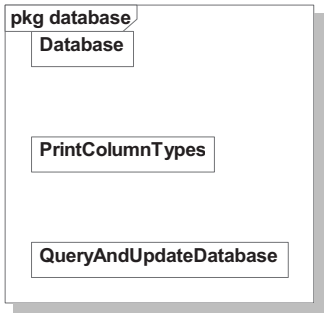


Gráfico 3.4 Paquete database

3.4.2.2. Paquete dataquery

Funciona en conjunto con el paquete *database*, realiza consultas más específicas sobre el contenido de la base datos, como la lista de estaciones de trabajo y puertos utilizados.

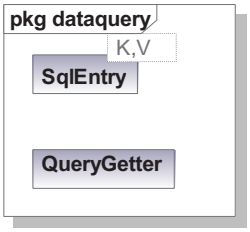


Gráfico 3.5 Paquete dataquery

3.4.2.3. Paquete datacapture

Sin duda es uno de los más importantes. Contiene clases con métodos para la apertura de captura del dispositivo de red, captura de paquetes, decodificación, almacenamiento y organización en memoria de los datos capturados, actualización de la base de datos, ejecución del sniffer, graficación en tiempo real total y diferenciación de tráfico.

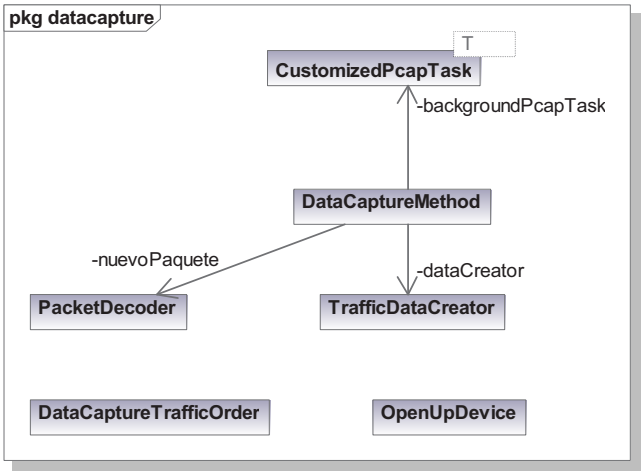


Gráfico 3.6 Paquete datacapture

3.4.2.4. Paquete descriptivestatistics

Paquete dedicado para el cálculo de parámetros de la estadística descriptiva.

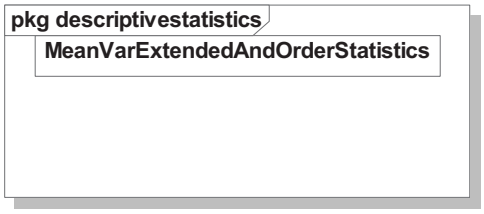


Gráfico 3.7 Paquete descriptivestatistics

3.4.2.5. Paquete dinamicstructures

Contiene las clases utilizadas por el paquete *datacapture* para la obtención de la lista dispositivos de red, para la organización en memoria del tráfico de red mediante listas dinámicas previa a la actualización de la base de datos.

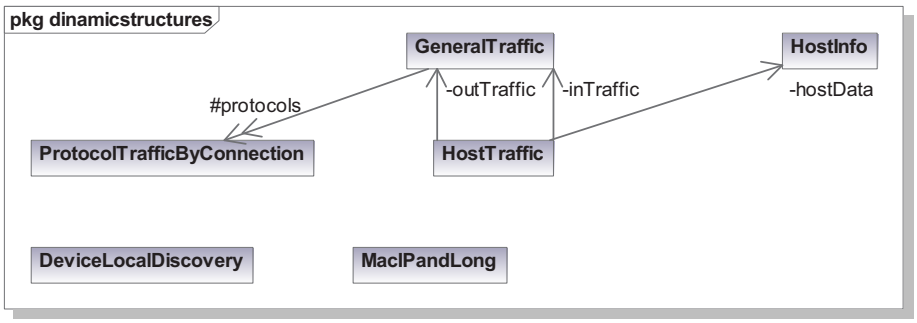


Gráfico 3.8 Paquete dinamicstructures

3.4.2.6. Paquete flashVideo

Paquete que permite la reproducción de animaciones flash y video para contenido de ayuda.

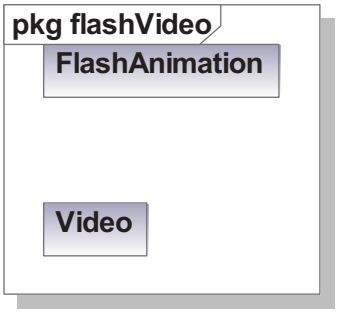


Gráfico 3.9 Paquete flashVideo

3.4.2.7. Paquete graphicsDialogs

Otro paquete muy importante, especialmente para el análisis estadístico. Contiene la implementación de diferentes cuadros de diálogo para cada uno de los resúmenes y gráficos estadísticos que deben ser generados, de acuerdo a los requerimientos a cumplirse. En general lo que se hace es generar una consulta a la base de datos, según las opciones elegidas por el usuario del programa, como pueden ser estaciones de trabajo seleccionadas, fecha de inicio y fin, hora de inicio y fin, protocolos y puertos.

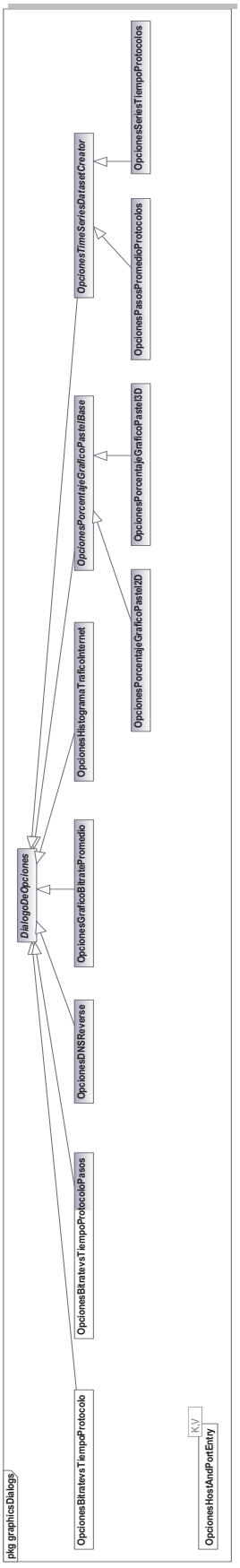


Gráfico 3.10 Paquete graphicsDialogs

3.4.2.8. Paquete lookandfeel

Paquete para mejoramiento visual de las aplicaciones Java, contiene clases con métodos para la selección de los distintos temas y almacenamiento en un fichero de la selección actual.

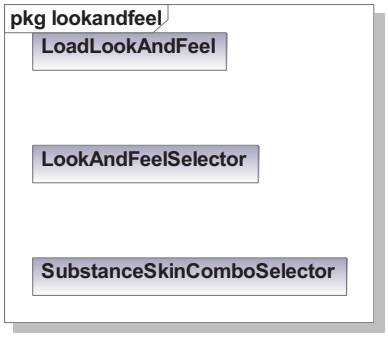


Gráfico 3.11 Paquete lookandfeel

3.4.2.9. Paquete importexportdatabase

Como su nombre indica, las clases contenidas en este paquete permiten exportar un respaldo total de la base de datos en un archivo zip y su posterior importación. La finalidad de esta implementación es la de proveer portabilidad de la base de datos, para que pueda ser trasladada de una estación de trabajo a otra, en caso de ser necesario.

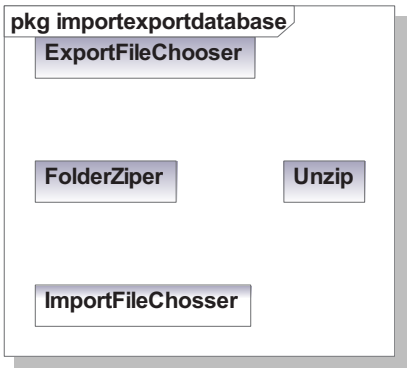


Gráfico 3.12 Paquete importexportdatabase

3.4.2.10. Paquete minibrowser

Implementa un componente gráfico que embebe al navegador web nativo, puede ser útil para el usuario en escenarios como la resolución de direcciones IP cuando se usa el sniffer.

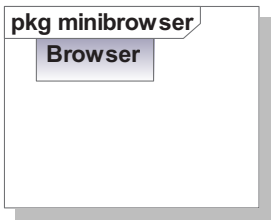


Gráfico 3.13 Paquete minibrowser

3.4.2.11. Paquete newhostdetected

Un requerimiento muy importante para este proyecto es la diferenciación de tráfico por estación de trabajo. Una forma de proveer soporte para esta necesidad es implementando un elemento gráfico que permita visualizar y elegir de entre el conjunto de estaciones de trabajo monitoreadas las que se desee. Este paquete personaliza un componente gráfico tipo árbol, para que cada estación de trabajo que sea añadida incluya una caja de selección (checkbox en inglés).

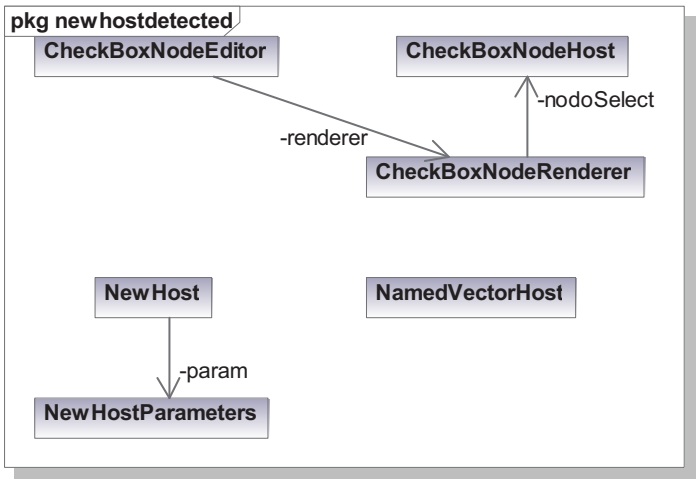


Gráfico 3.14 Paquete newhostdetected

3.4.2.12. Paquete realtimegraphs

Graficación del tráfico de Internet, ese es el objetivo de las clases contenidas en este paquete. Incluyen los mecanismos necesarios para la actualización automática del gráfico, ajuste automático de la escala del gráfico de acuerdo al valor pico máximo, zoom del área seleccionada, guardado de la imagen, gradiente de color y alarma sonora para identificar escenarios críticos en base a un valor de tasa de transferencia suministrado por el usuario, etiquetas de texto con los valores de la tasa de transferencia instantánea y promedio, reinicio y finalización.

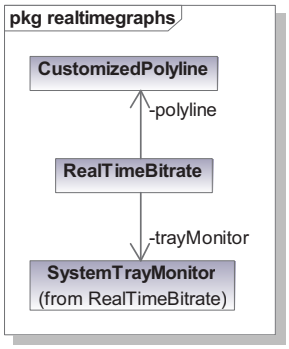


Gráfico 3.15 Paquete realtimegraphs

3.4.2.13. Paquete reversednsresolver

Paquete complementario que permite realizar la resolución inversa de nombres dada una dirección IP, en caso de que existan registros en los servidores DNS ingresados.

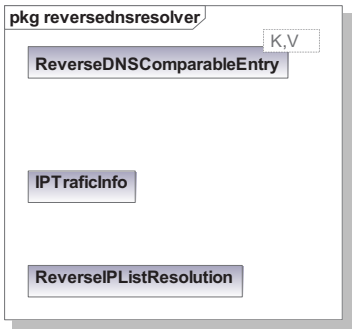


Gráfico 3.16 Paquete reversednsresolver

3.4.2.14. Paquete save

Contiene clases que implementan métodos para guardar imágenes y texto, cuadros de diálogo y autoguardado del contenido de un área de texto.

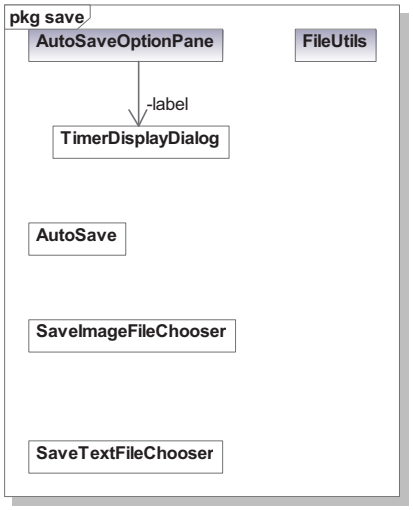


Gráfico 3.17 Paquete save

3.4.2.15. Paquete sniffer

Paquete que implementa la funcionalidad de un sniffer, incluida la capacidad de seleccionar filtros para protocolos, puertos y estaciones de trabajo, decodificación total del paquete o sólo de sus cabeceras de protocolo en formato de texto, graficación del tráfico entrante y saliente para estas estaciones.

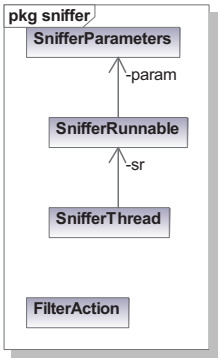


Gráfico 3.18 Paquete sniffer

3.4.2.16. Paquete statisticalgraphics

Este paquete implementa todos los gráficos y resúmenes informativos para el tráfico de Internet, especificados en el capítulo de requerimientos. Trabaja conjuntamente con el paquete graphicsDialogs, este último recopila las opciones elegidas por el usuario y las clases de statisticalgraphics se encargan de la generación de los gráficos y resúmenes. Incluyen todas las utilidades para guardar texto y gráficos.

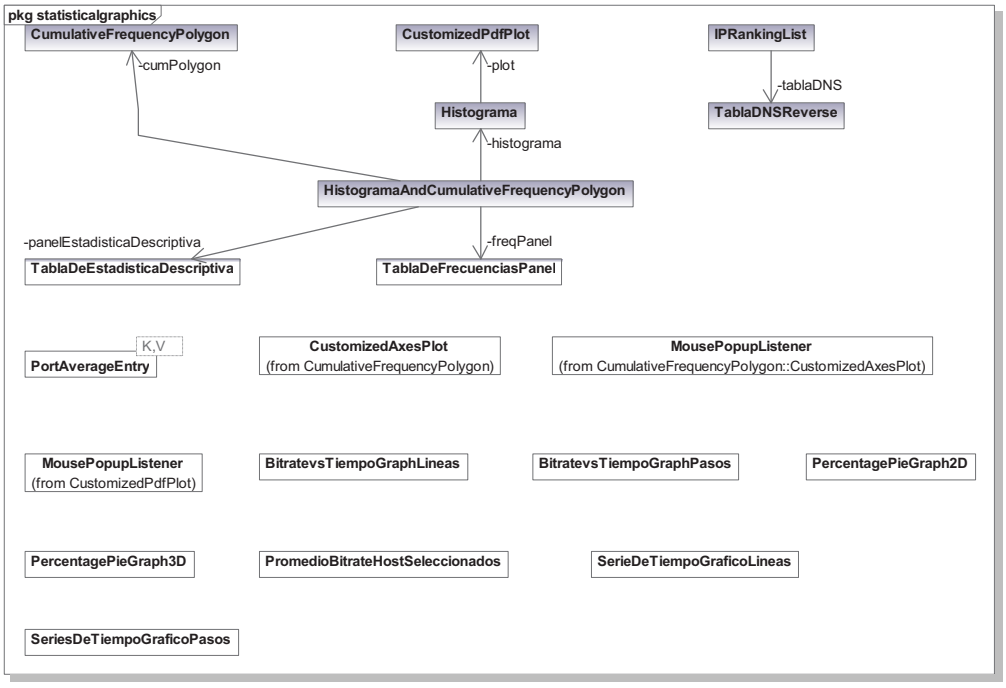


Gráfico 3.19 Paquete statisticalgraphics

3.4.2.17. Paquete sugerencia

Conjunto de clases para el despliegue de páginas web con contenido de sugerencias y ayuda para la interpretación de los gráficos generados por el paquete statisticalgraphics.

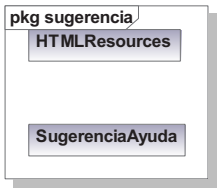


Gráfico 3.20 Paquete sugerencia

3.4.3. DESCRIPCIÓN DE CLASES Y DIAGRAMAS DE ACTIVIDAD

A continuación se describirán las clases de los distintos paquetes en los que se han distribuido en la biblioteca principal. Dada la extensión del proyecto, los diagramas de actividad se mostrarán para los métodos que se consideran más relevantes, desde el punto de vista de mayor complejidad o funcionalidad. Para mayor información sobre el significado e interpretación de estos diagramas se recomienda leer el anexo A.

3.4.3.1. Clases del paquete database

3.4.3.1.1. Clase Database

La clase Database permite la creación, inicialización y finalización de la base de datos desde la máquina virtual de Java. Además contiene métodos para obtener la conexión actual, un ejecutor de sentencias SQL y un verificador de existencia de la base de datos.

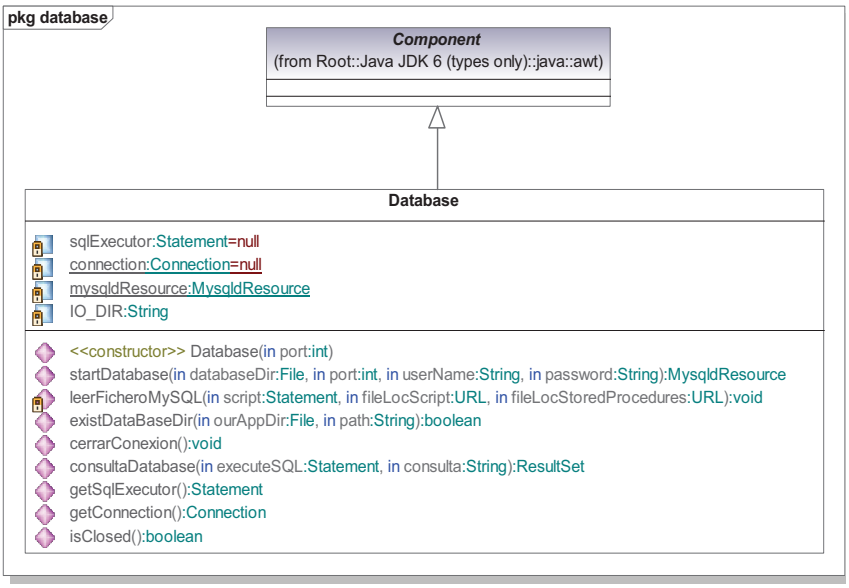


Gráfico 3.21 Clase Database

El gráfico 3.22 muestra las actividades realizadas cuando se instancia un objeto Database.

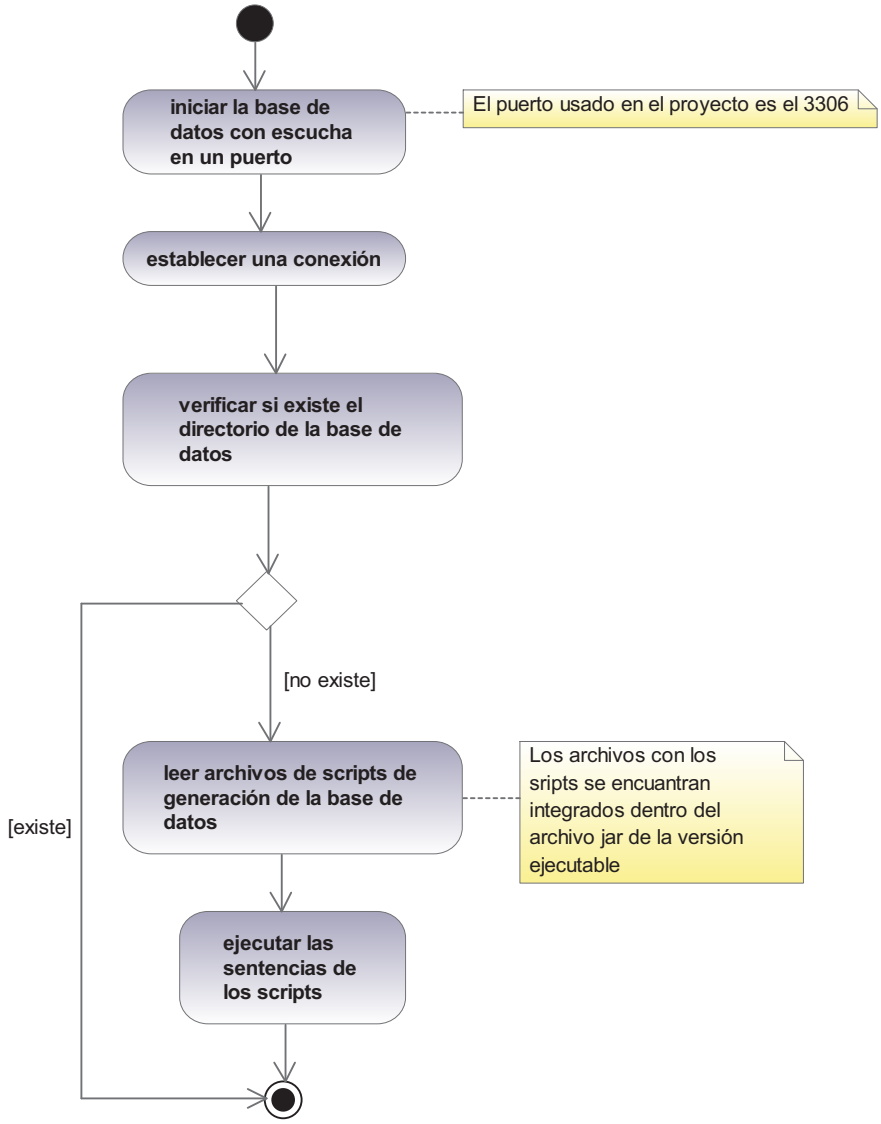


Gráfico 3.22 Diagrama de actividad del constructor de la clase Database

3.4.3.1.2. Clase PrintColumnTypes

La clase PrintColumnTypes permite imprimir los tipos de datos obtenidos correspondientes a las columnas en una tabla de una consulta.

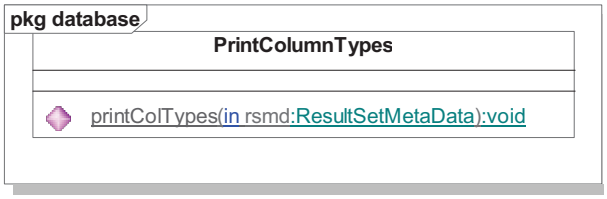


Gráfico 3.23 Clase PrintColumnTypes

3.4.3.1.3. Clase QueryAndUpdateDatabase

La clase QueryAndUpdateDatabase permite actualizar la base de datos agregando la cantidad en bytes de datos enviados o recibidos por conexión, para cada estación de trabajo, además realiza consultas a la misma para devolver el identificador de cada host dentro de los registros almacenados. Permite añadir nuevas estaciones de trabajo con sus respectivos datos.

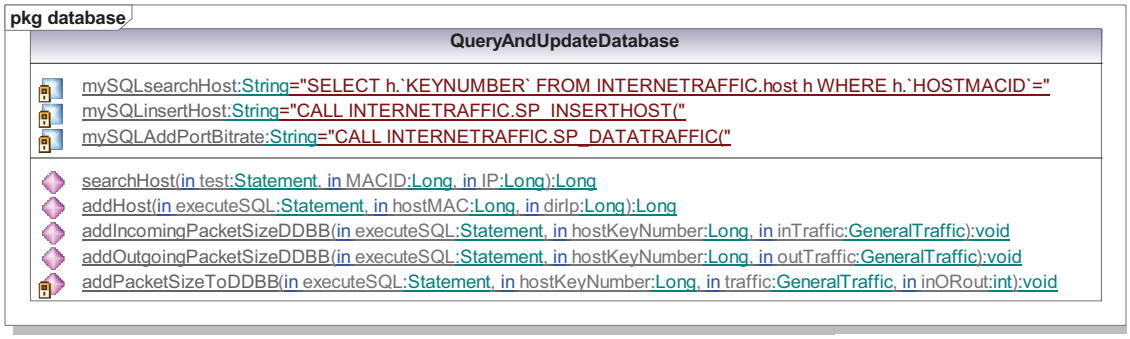


Gráfico 3.24 Clase QueryAndUpdateDatabase

3.4.3.2. Clases del paquete dataquery

3.4.3.2.1. Clase QueryGetter

La clase QueryGetter permite inicializar y manejar la base de datos con un puerto de conexión determinado, delega algunos métodos de la clase Database e implementa algunos propios. Esto incluye métodos para realizar consultas específicas, como por ejemplo, la lista de las estaciones de trabajo presentes en la base de datos, una lista de los puertos de capa transporte registrados para las estaciones seleccionadas, en un rango de tiempo dado.

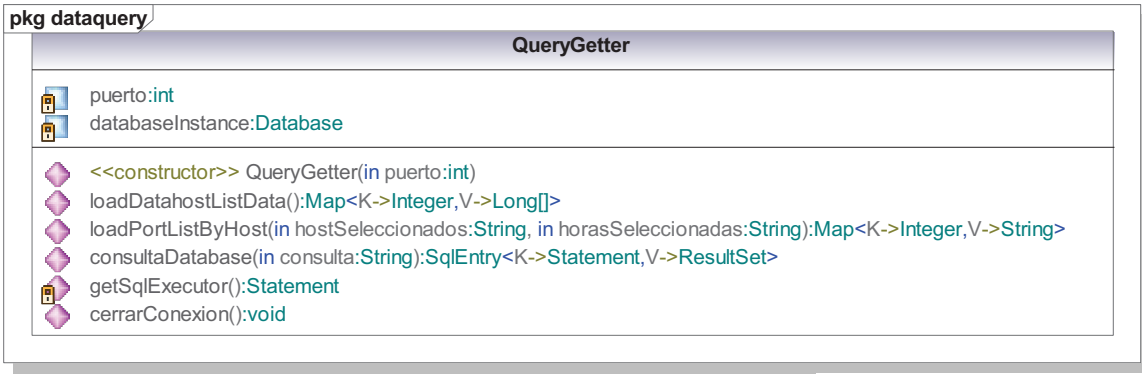


Gráfico 3.25 Clase QueryGetter

3.4.3.2.2. Clase SqlEntry

Una instancia de la clase SqlEntry mantiene una relación unívoca entre un objeto Statement y un objeto ResultSet.

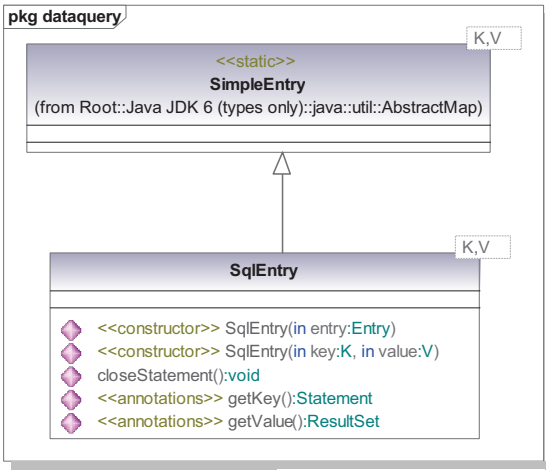


Gráfico 3.26 Clase SqlEntry

3.4.3.3. Clases del paquete datacapture

3.4.3.3.1. Clase CustomizedPcapTask

La clase CustomizedPcapTask es un manejador de las tareas de captura de un objeto Pcap, que corre a nivel de background y provee métodos de control y estado sobre el loop. A diferencia de PcapTask (clase de la cual realiza herencia), esta incluye un mecanismo seguro para detener el hilo de ejecución.

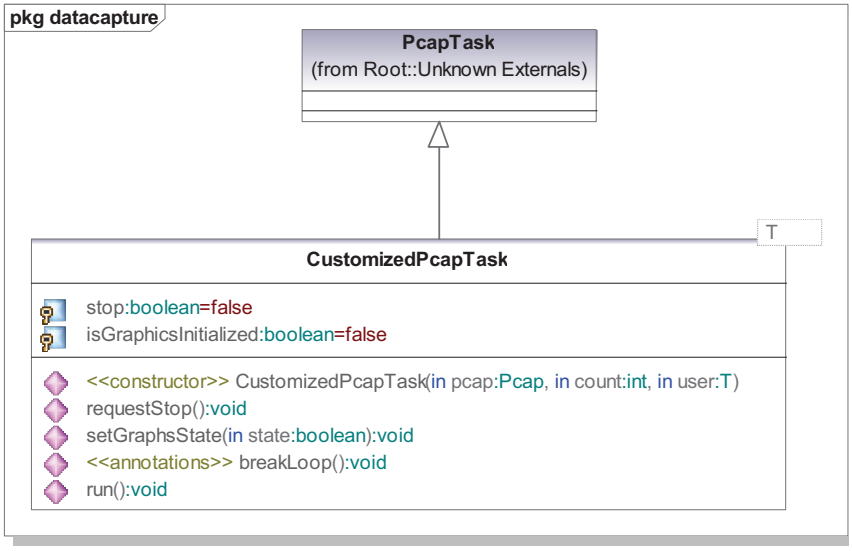


Gráfico 3.27 Clase CustomizedPcapTask

3.4.3.3.2. Clase DataCaptureMethod

La clase DataCaptureMethod permite el control total del monitoreo de una red con acceso a Internet, a través de métodos que interactúan con otras clases para proveer de acceso y almacenamiento automático de los datos capturados, de un sniffer con todas sus funcionalidades comunes, graficación de tráfico en tiempo real y gestión de las estructuras temporales de los paquetes capturados previa a la actualización de la base de datos.

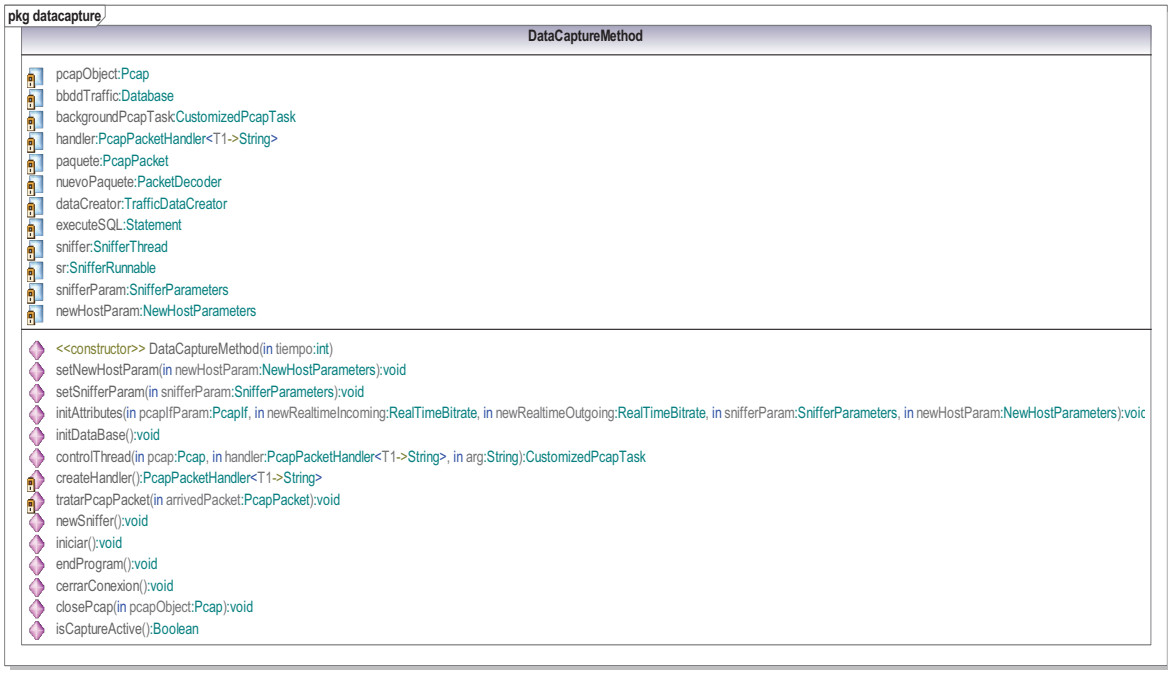


Gráfico 3.28 Clase DataCaptureMethod

Para controlar la captura de paquetes DataCaptureMethod sobrescribe el método run() de un objeto CustomizedPcapTask para que realice las siguientes actividades, como se muestran en el gráfico 3.29.

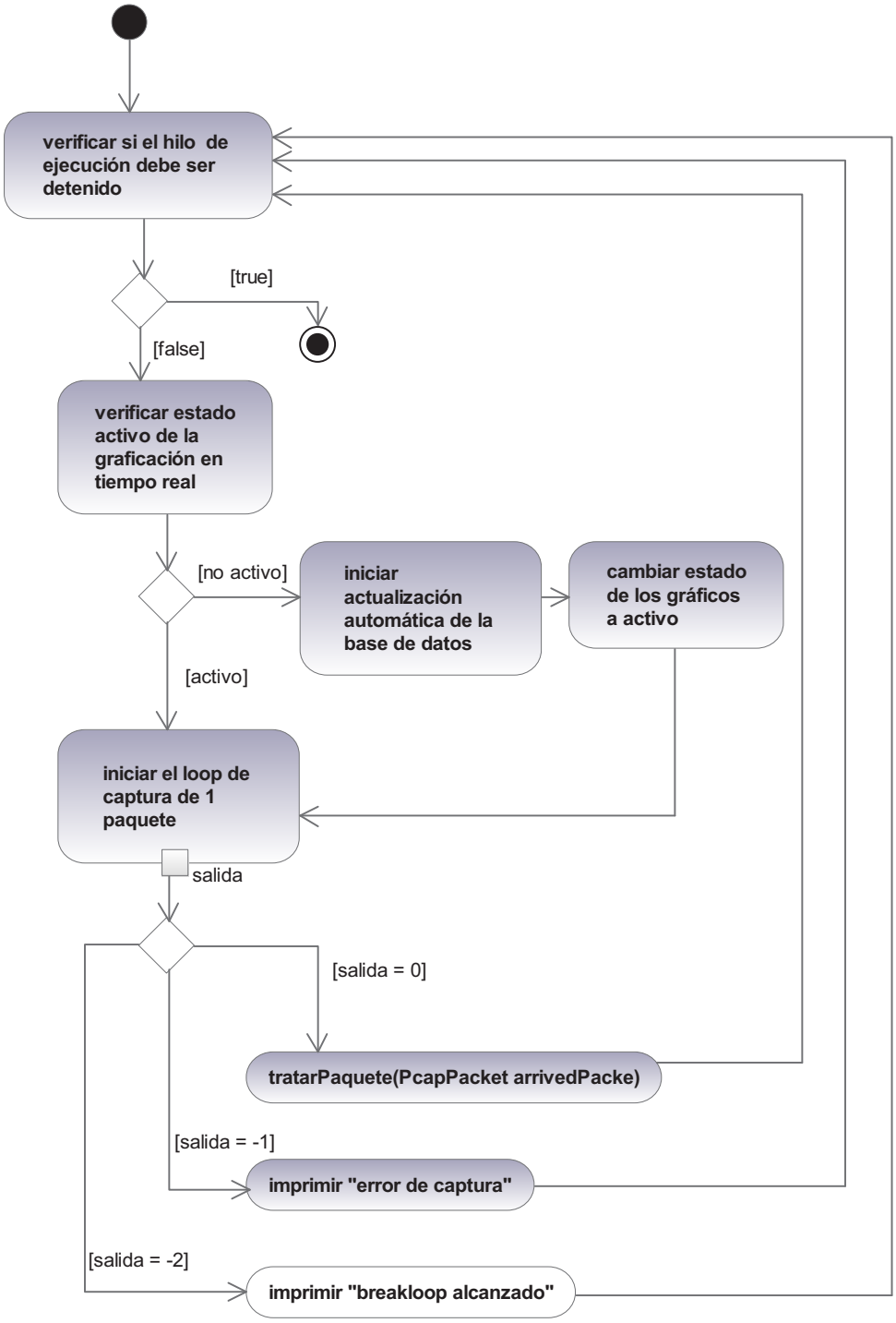


Gráfico 3.29 Diagrama de actividad del método run() del hilo de ejecución de captura de paquetes

En el gráfico anterior se observa que el hilo de captura de paquetes seguirá ejecutándose, hasta que externamente se solicite la finalización de la ejecución. También puede observarse que cuando el paquete se ha capturado con éxito, el método tratarPaquete(…) es utilizado.

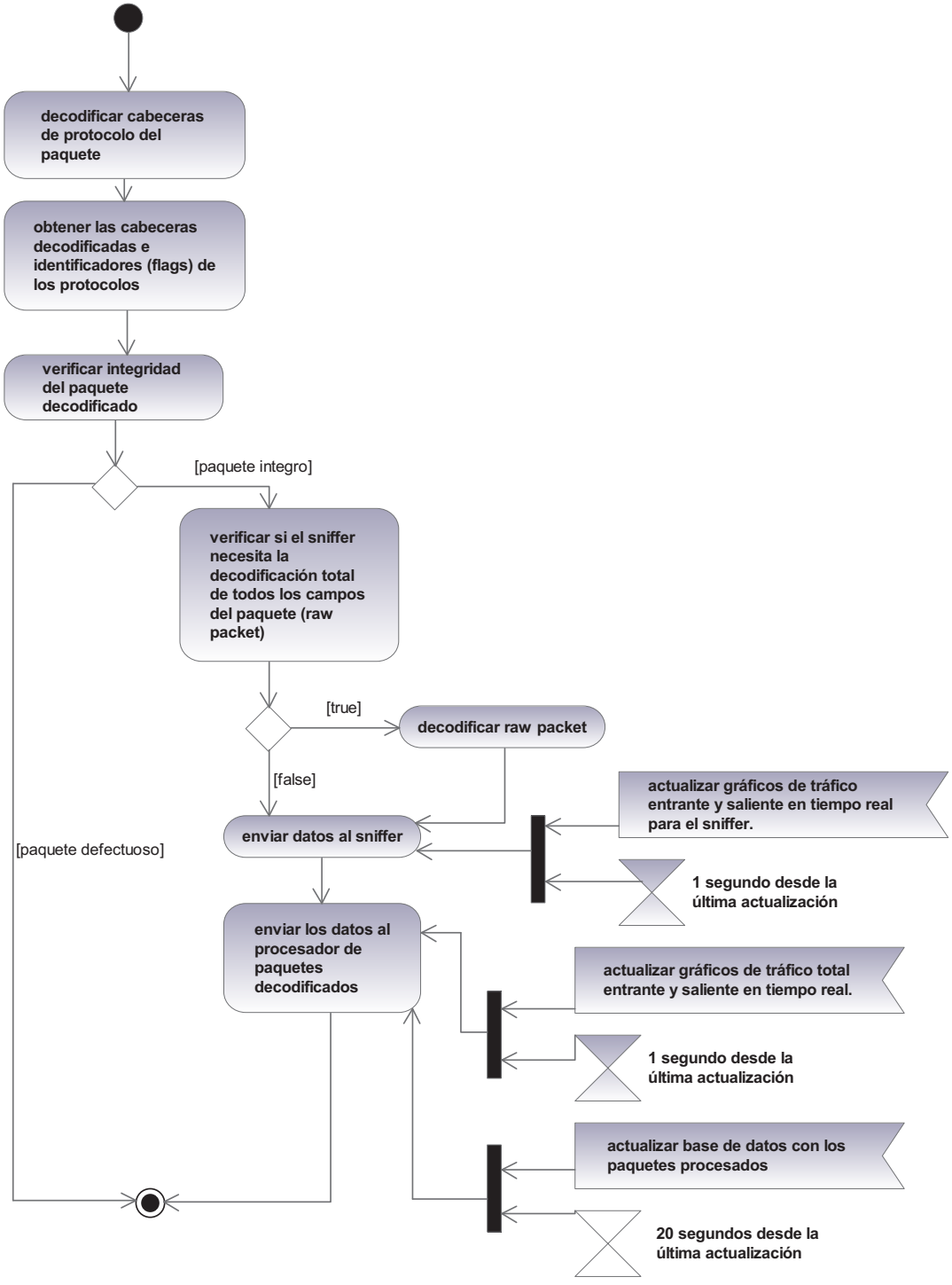


Gráfico 3.30 Diagrama de actividad del método tratarPaquete(…) de la clase DataCaptureMethod

3.4.3.3.3. Clase DataCaptureTrafficOrder

La clase DataCaptureTrafficOrder permite clasificar e identificar el tráfico entrante y el tráfico saliente por medio de las direcciones IP del paquete. También incluye un método para determinar si una dirección IP es privada, tomando en cuenta los casos especiales de direcciones de loopback y broadcast.

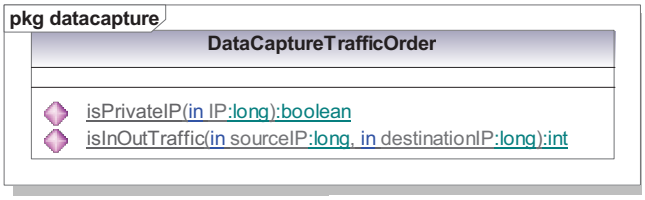


Gráfico 3.31 Clase DataCaptureTrafficOrder

3.4.3.3.4. Clase OpenUpDevice

La clase OpenUpDevice establece un método que fija los parámetros de captura de información de los paquetes, como son: el máximo número de bytes a capturar, el tiempo de captura de datos antes de devolver los paquetes a la aplicación, el modo de captura (en este caso MODE_PROMISCUOUS).

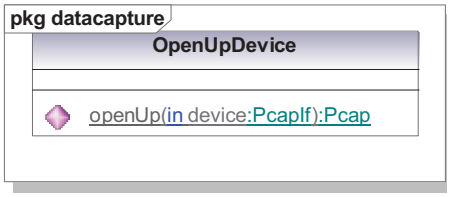


Gráfico 3.32 Clase OpenUpDevice

A continuación un diagrama de actividades que detalla este método.

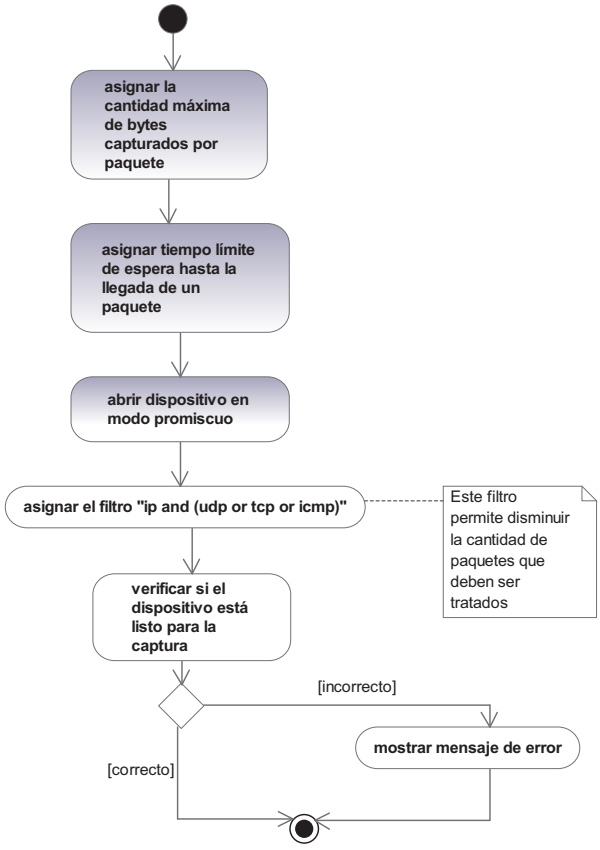


Gráfico 3.33 Diagrama de actividad del método OpenUp() de la clase OpenUpDevice

3.4.3.3.5. Clase PacketDecoder

La clase PacketDecoder decodifica y verifica las cabeceras de protocolo del paquete para hacer una distinción del tráfico existente. Retorna un array de objetos con los valores de los campos de protocolo decodificados. Incluye un método para crear una representación completa en forma de cadena de texto de todos los campos del paquete capturado (raw packet).

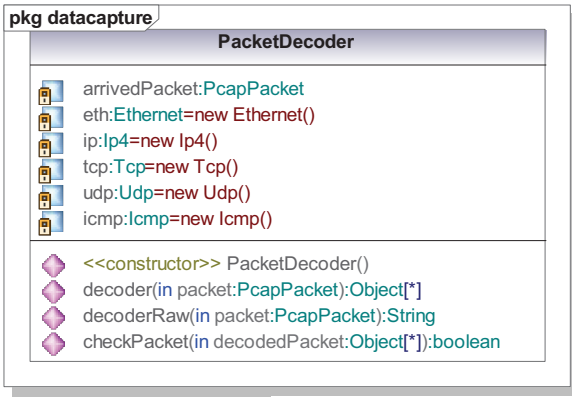


Gráfico 3.34 Clase PacketDecoder

3.4.3.3.6. Clase TrafficDataCreator

La clase TrafficDataCreator trata el paquete capturado organizando la información para cada estación de trabajo, tomando en cuenta el tráfico entrante, el tráfico saliente, protocolos, el tamaño del paquete y puertos de origen y destino. Actualiza periódicamente la base de datos y los gráficos correspondientes al bitrate total en tiempo real, ejecutando hilos independientes.

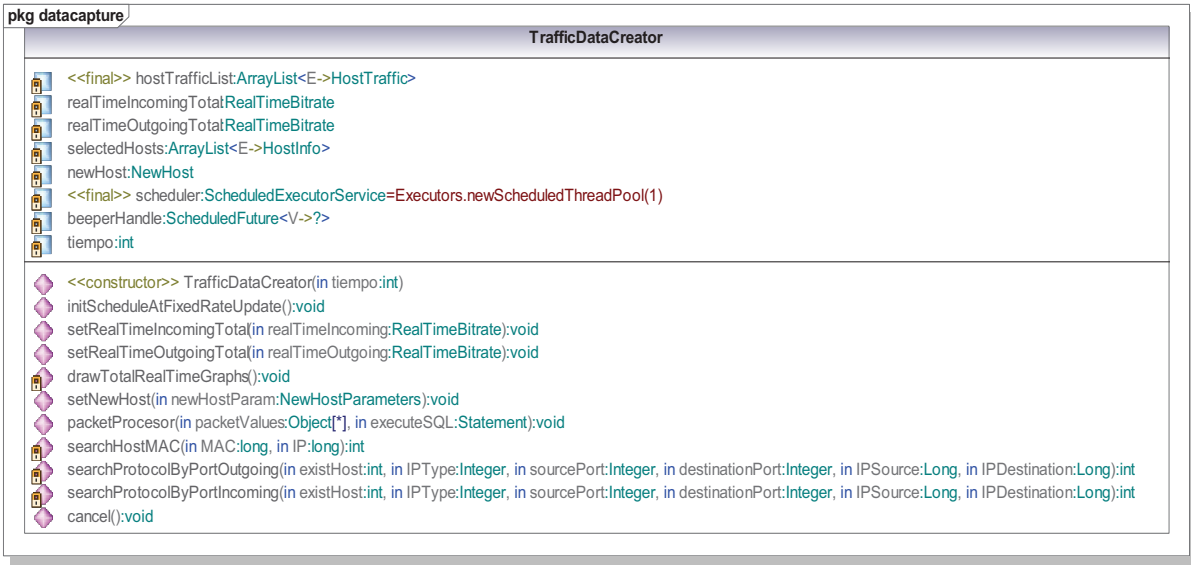


Gráfico 3.35 Clase TrafficDataCreator

El método initScheduleAtFixedRateUpdate() inicia dos actividades periódicas y automáticas que se detallan en el gráfico 3.36.

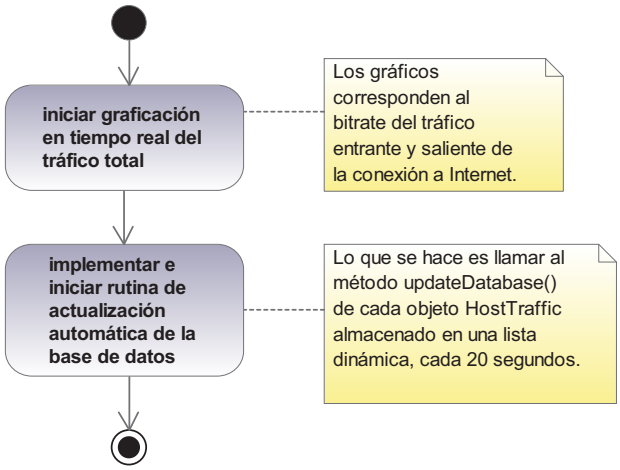


Gráfico 3.36 Diagrama de actividad del método initScheduleAtFixedRateUpdate() de la clase TrafficDataCreator

Otro método muy importante es packetProcesor(…,….) que engloba casi todas funcionalidades descritas en la descripción de la clase.

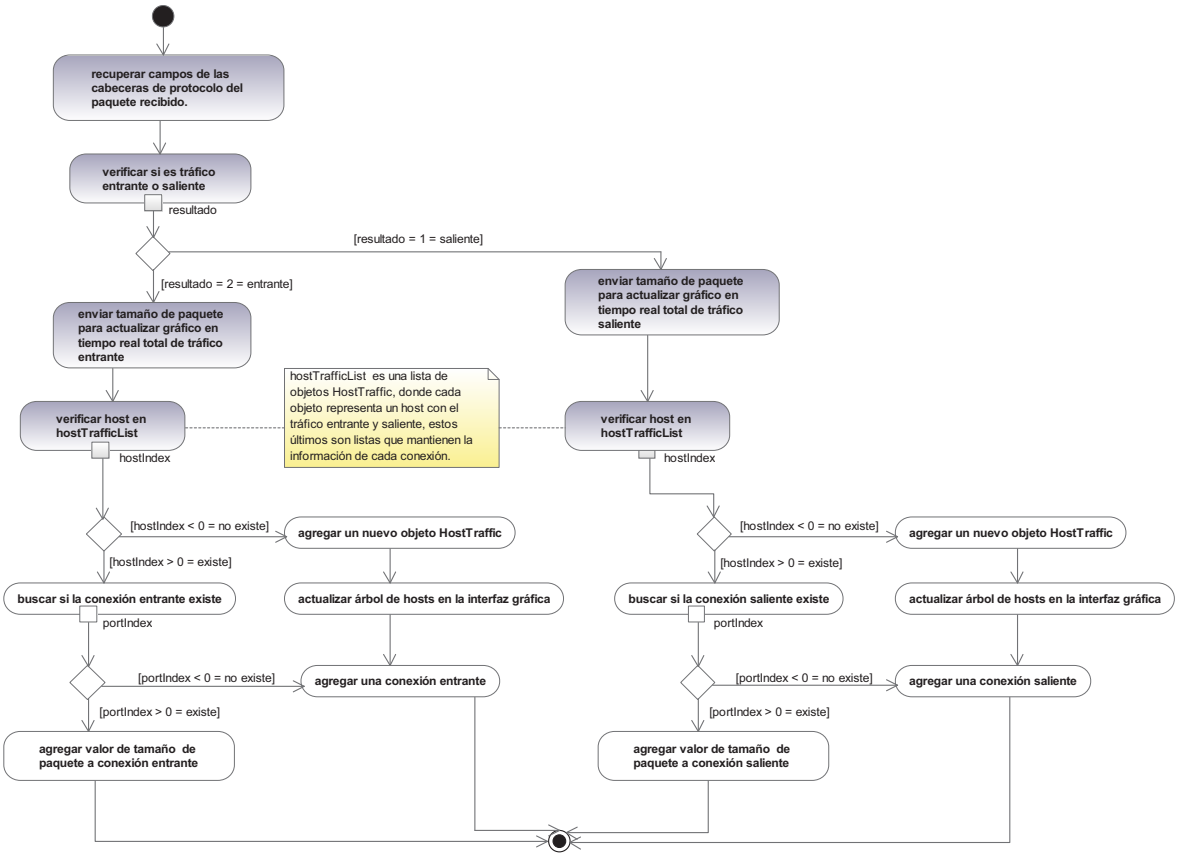


Gráfico 3.37 Diagrama de actividad del método packetProcesor (…,…) de la clase TrafficDataCreator

3.4.3.4. Clases del paquete descriptivestatistics

3.4.3.4.1. Clase MeanVarExtendedAndOrderStatistics

La clase MeanVarExtendedAndOrderStatistics permite calcular de un array de valores double, todos los parámetros necesarios para mostrar un resumen de Estadística Descriptiva. Para ello hereda de MeanVar y tiene un atributo de tipo OrderStatistics (Ambas clases del paquete estadístico JSC 1.0).

Los más importantes son: la media, la varianza, el valor máximo, el valor mínimo, los cuartiles, la media recortada, el rango y el error estándar de la media.

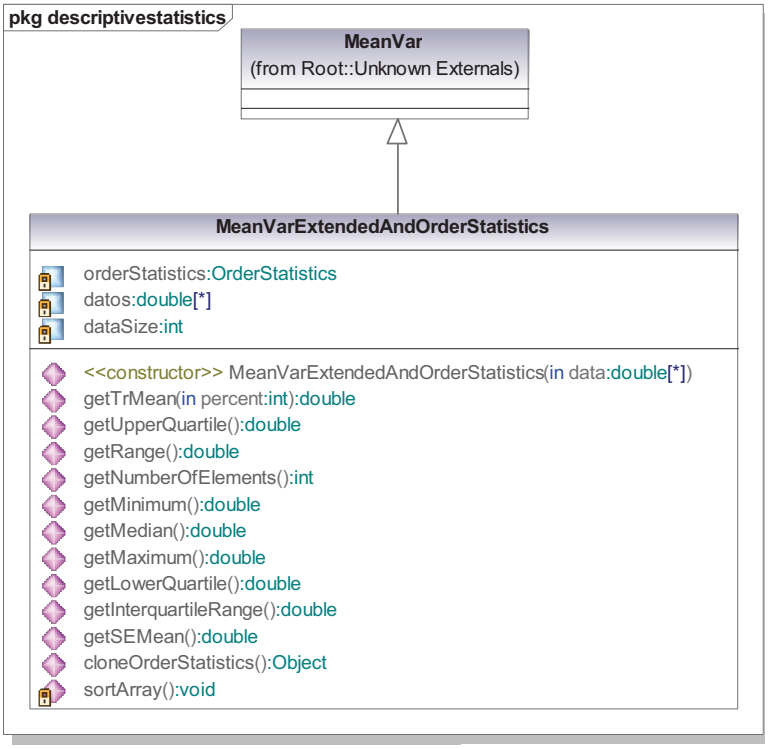


Gráfico 3.38 Clase MeanVarExtendedAndOrderStatistics

3.4.3.5. Clases del paquete dinamicstructures

3.4.3.5.1. Clase DeviceLocalDiscovery

Clase que implementa el descubrimiento de dispositivos de red activos en la PC actual.

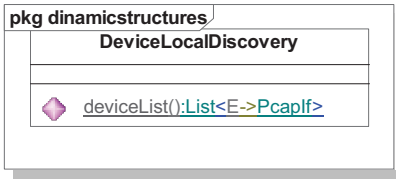


Gráfico 3.39 Clase DeviceLocalDiscovery

3.4.3.5.2. Clase GeneralTraffic

La clase GeneralTraffic contiene una lista de todos los protocolos por conexión, que pueden ser de tráfico entrante o saliente, de manera indistinta.

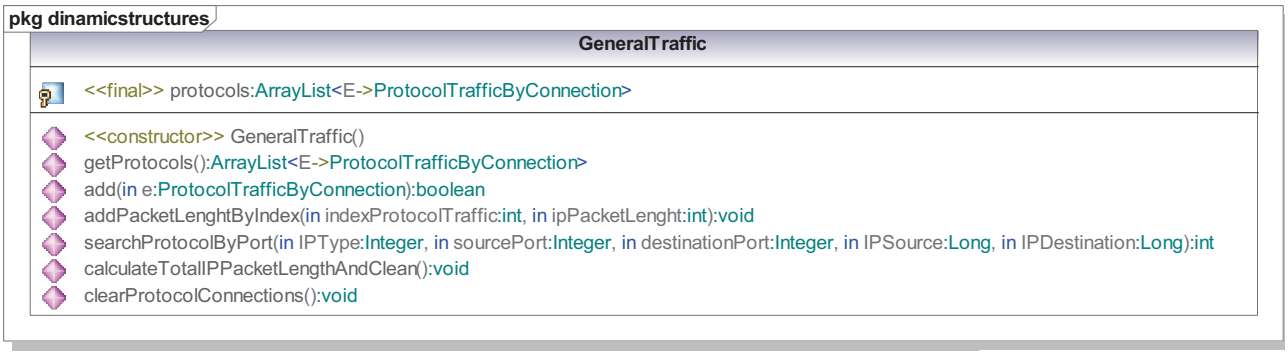


Gráfico 3.40 Clase GeneralTraffic

3.4.3.5.3. Clase HostInfo

La clase HostInfo permite abstraer una estación de trabajo con fines de identificación para el monitoreo de tráfico de Internet, con una dirección MAC y una dirección IP.

Gráfico 3.41 Clase HostInfo

3.4.3.5.4. Clase HostTraffic

La clase HostTraffic asigna a una estación de trabajo dos atributos que son tráfico entrante y tráfico saliente, cada uno en un objeto GeneralTraffic, donde se organiza la información para cada conexión activa, además contiene un método para ingresar a la base de datos una nueva estación de trabajo y realizar una actualización de sus datos.

Gráfico 3.42 Clase HostTraffic

El método con la funcionalidad más crítica es sin duda updateDatabase(). Este método es ejecutado cada 20 segundos por un hilo de ejecución independiente y se detalla en el diagrama de actividad del gráfico 3.43.

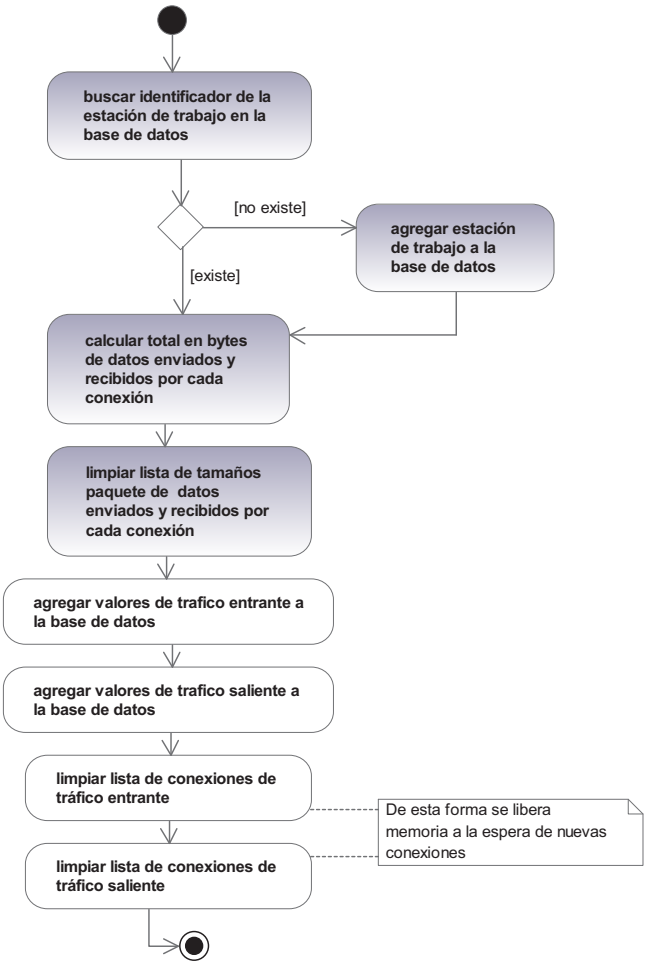


Gráfico 3.43 Diagrama de actividad del método updateDatabase() de la clase HostTraffic

3.4.3.5.5. Clase MacIPandLong

La clase MacIPandLong contiene a un conjunto de métodos que permiten trabajar con las direcciones IP y el valor de la MAC del dispositivo de red. Estos métodos fueron pensados para trabajar con valores numéricos que representan a las direcciones IP y la MAC, además de sus correspondientes representaciones en forma de cadena de caracteres.

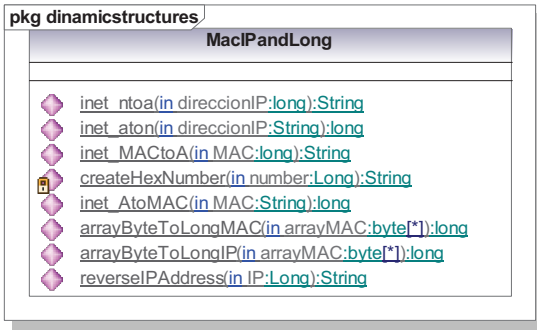


Gráfico 3.44 Clase MacIPandLong

3.4.3.5.6. Clase ProtocolTrafficByConnection

La clase ProtocolTrafficByConnection identifica una conexión por medio de sus atributos, que permiten diferenciar una de otra por cada protocolo de capa transporte y por estación de trabajo. Se almacenan una lista de los tamaños de los paquetes capturados en cada conexión, tomando en cuenta todo el paquete IP con su cabecera.

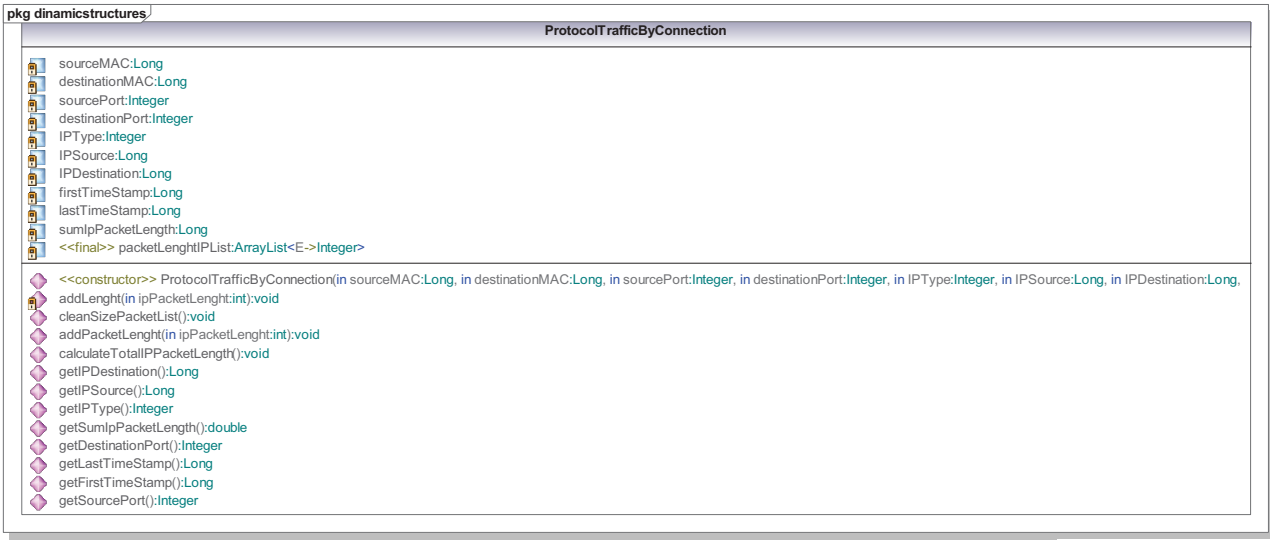


Gráfico 3.45 Clase ProtocolTrafficByConnection

3.4.3.6. Clases del paquete flashVideo

3.4.3.6.1. Clase FlashAnimation

La clase FlashAnimation permite cargar y manejar archivos con extensión .swf desde un path especificado, a través de un reproductor flash nativo.

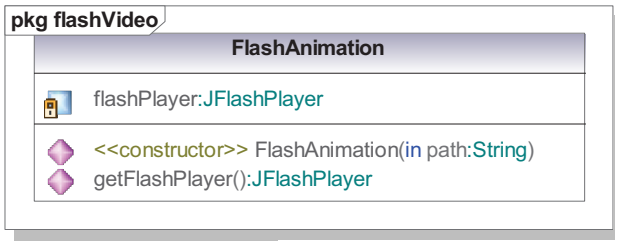


Gráfico 3.46 Clase FlashAnimation

3.4.3.6.2. Clase Video

La clase Video permite utilizar el reproductor de video nativo del sistema operativo, así como un selector de archivos de video y una casilla de activación que permite ocultar la barra de control del reproductor.

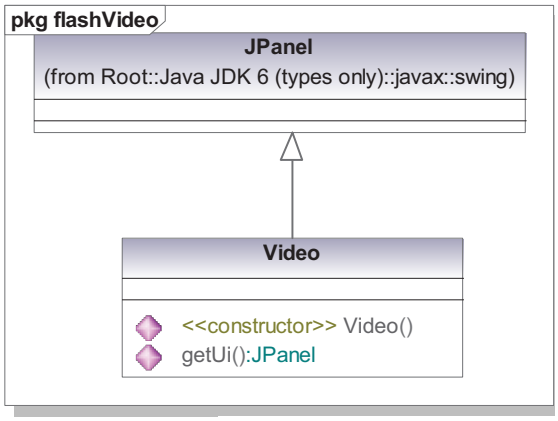


Gráfico 3.47 Clase Video

3.4.3.7. Clases del paquete graphicsDialogs

3.4.3.7.1. Clase DialogoDeOpciones

La clase DialogoDeOpciones utiliza un cuadro de diálogo que contiene todos los parámetros necesarios para armar una consulta SQL a la base de datos y obtener una tabla con resultados útiles, para poder generar cada uno de los diagramas. Esto incluye un selector de estaciones de trabajo, protocolos, fecha (por medio de calendarios gráficos) e intervalo de horas y puertos. Esta clase abstracta es la

base para todos los diálogos de opciones que heredan su funcionalidad e implementan los métodos de generación de los gráficos de manera particular.

Para el diseño de esta clase se utilizó la herramienta gráfica de diseño de NetBeans IDE 6.5 con los elementos de la paleta de componentes gráficos. Esto simplifica el grado de abstracción necesario en la elaboración de una interfaz gráfica.

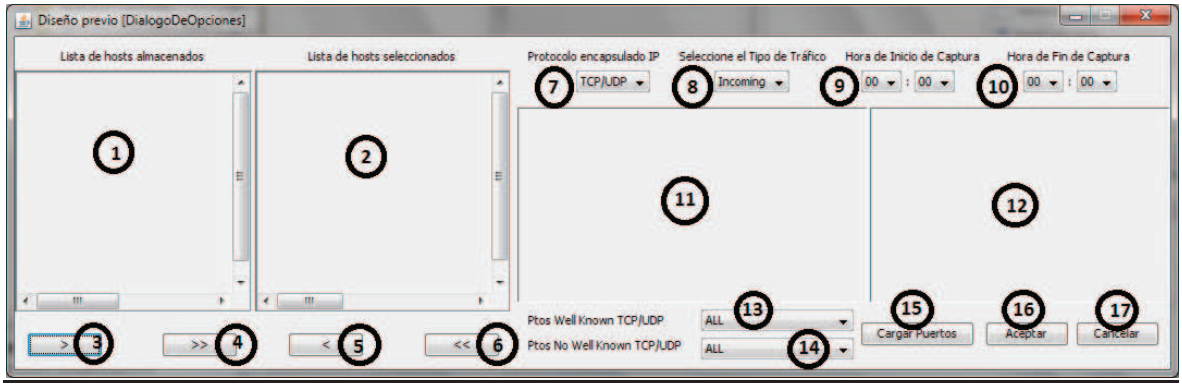


Gráfico 3.48 Vista previa de la ventana de DialogoDeOpciones

Identificador	Componente	Descripción
1	JPanel	Panel donde se alojará una lista de las estaciones de trabajo existentes en la base de datos.
2	JPanel	Panel donde se ubicará la lista de las estaciones de trabajo seleccionadas.
3	JButton	Botón para enviar a una estación de trabajo a la lista de hosts seleccionados.
4	JButton	Botón para elegir todas las estaciones de trabajo.
5	JButton	Botón que quita una estación de trabajo de la lista de hosts seleccionados.
6	JButton	Botón que vacía la lista de hosts seleccionados.
7	JComboBox	Lista desplegable para selección de protocolos: TCP/UDP, TCP, UDP, ICMP, ALL (todos sin restricción).
8	JComboBox	Lista desplegable para seleccionar tráfico entrante o tráfico saliente.
9	JComboBox	Dos listas desplegables para elegir la hora y el minuto del día desde el cual se recuperarán los datos.
10	JComboBox	Dos listas desplegables para elegir la hora y el minuto día tope para la consulta.
11	JPanel	Panel donde se ubicará un calendario visual para seleccionar la fecha del día de inicio.
12	JPanel	Panel donde se ubicará un calendario visual para seleccionar la fecha del día final.
13	JComboBox	Lista desplegable para elegir todos, ninguno o un puerto conocido específico del tráfico TCP/UDP.
14	JComboBox	Lista desplegable para elegir todos, ninguno o un puerto no conocido específico del tráfico TCP/UDP.
15	JButton	Botón que llena las listas desplegables de los puertos bien conocidos y no conocidos, tomando como parámetros las fechas de inicio y fin, las horas de inicio y fin, y las estaciones de trabajo seleccionadas.
16	JButton	Crea la sentencia para la consulta a la base y despliega el gráfico y los resúmenes de datos.
17	JButton	Cierra la ventana de diálogo.

Tabla 3.2 Descripción de la vista previa de la ventana de DialogoDeOpciones

Gráfico 3.49 Clase DialogoDeOpciones

Como puede verse en la vista previa de DialogoOpciones existe el botón Aceptar. Cuando este botón recibe un evento como un clic se ejecuta el método generarConsultaGrafico() que se detalla en el gráfico 3.50.

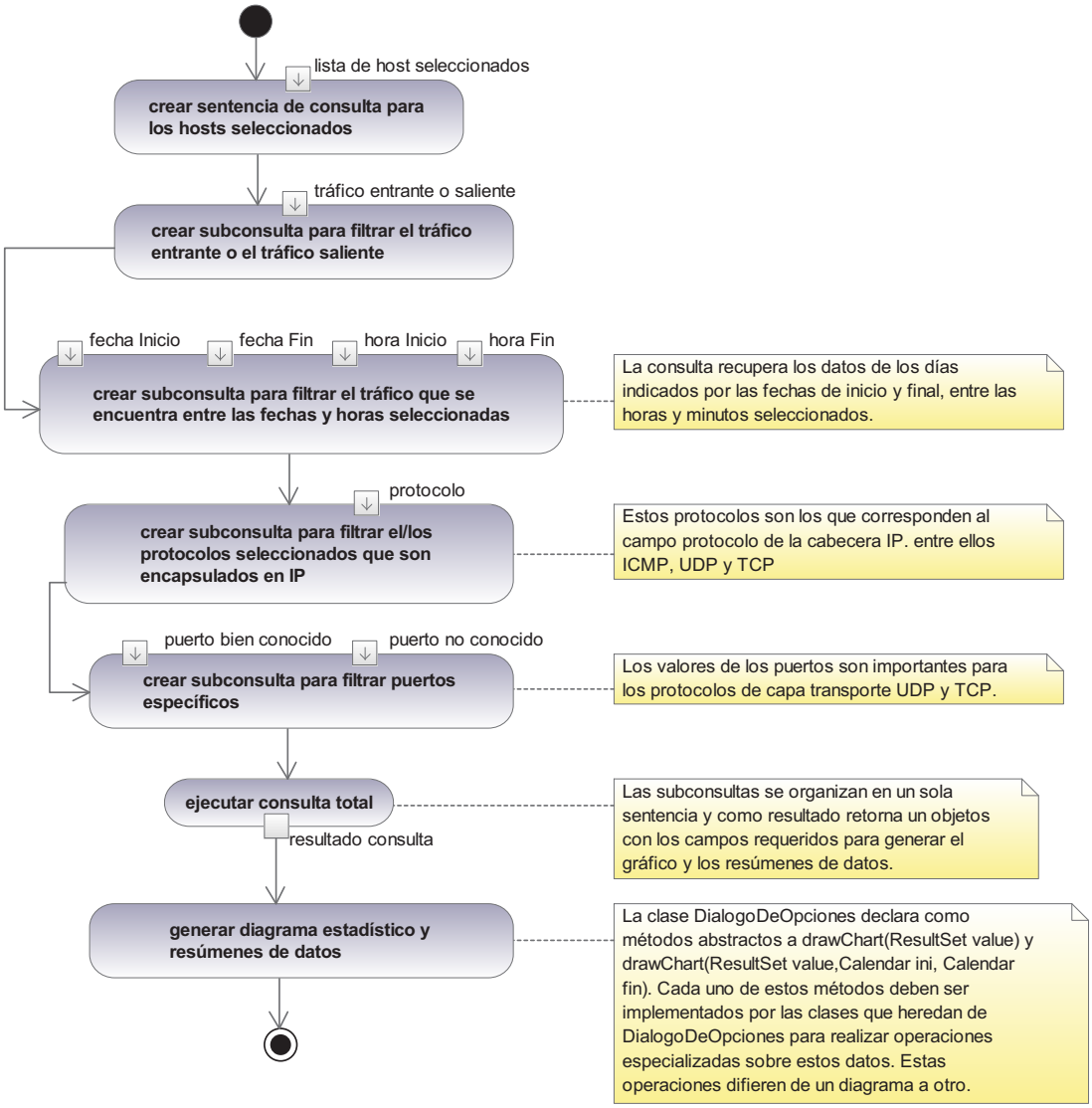


Gráfico 3.50 Diagrama de actividad del método generarConsultaGrafico() de la clase DialogoDeOpciones

3.4.3.7.2. Clase OpcionesHostAndPortEntry

La clase OpcionesHostAndPortEntry es un objeto que mantiene asociada una clave y su valor correspondiente. En el presente caso se la usa para almacenar pares de objetos relacionados, por ejemplo, el identificador de la estación de trabajo en la base de datos (clave) con la MAC e IP (valor) o un puerto (clave) y sus siglas (valor). También se implementa la funcionalidad de comparación para poder ordenar una lista dinámica de estos elementos posteriormente.

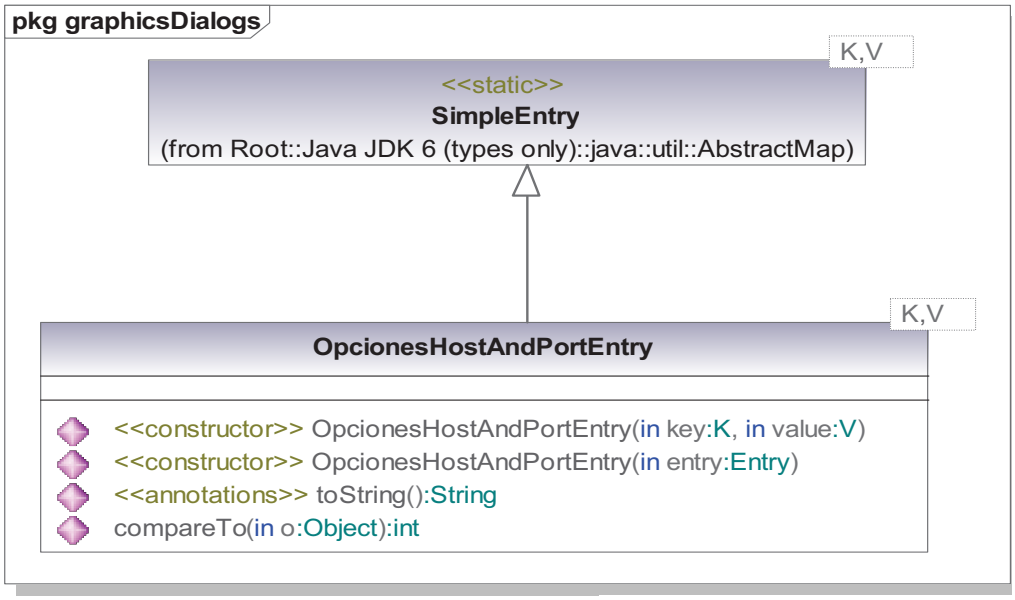


Gráfico 3.51 Clase OpcionesHostAndPortEntry

3.4.3.7.3. Clase OpcionesBitratevsTiempoProtocolo

La clase OpcionesBitratevsTiempoProtocolo muestra un cuadro de diálogo para obtener los valores necesarios para la graficación del diagrama de bitrate vs tiempo. Para ello hereda la funcionalidad de DialogoDeOpciones e implementa los métodos abstractos de manera personalizada.

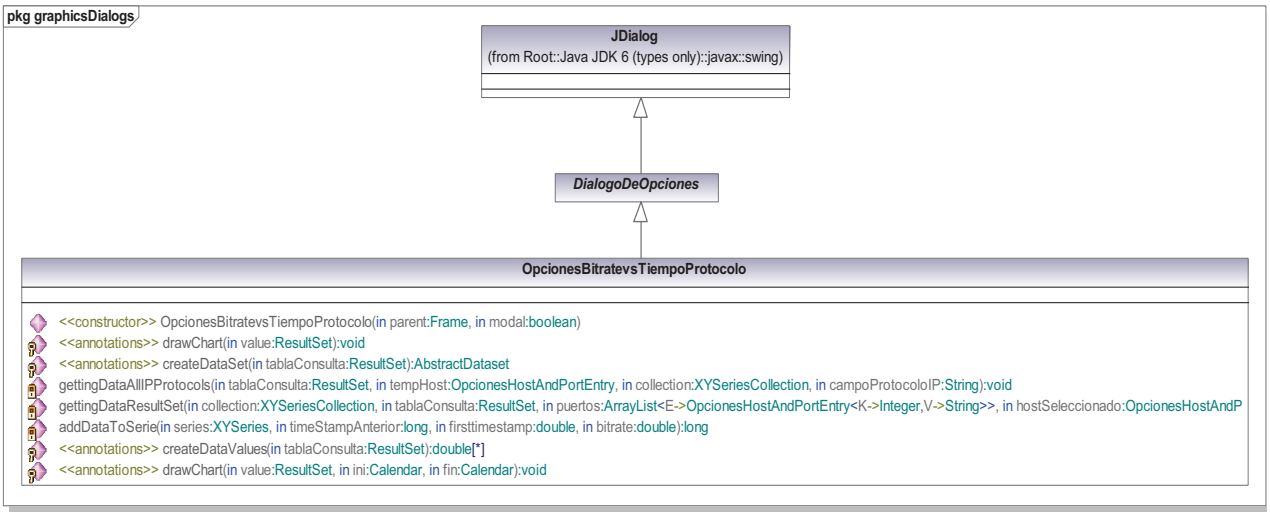


Gráfico 3.52 Clase OpcionesBitratevsTiempoProtocolo

3.4.3.7.4. Clase OpcionesBitratevsTiempoProtocoloPasos

La clase OpcionesBitratevsTiempoProtocoloPasos implementa los métodos abstractos de DialogoDeOpciones para obtener los datos de la base necesarios y graficar el diagrama de bitrate vs tiempo en forma de una serie de pasos.

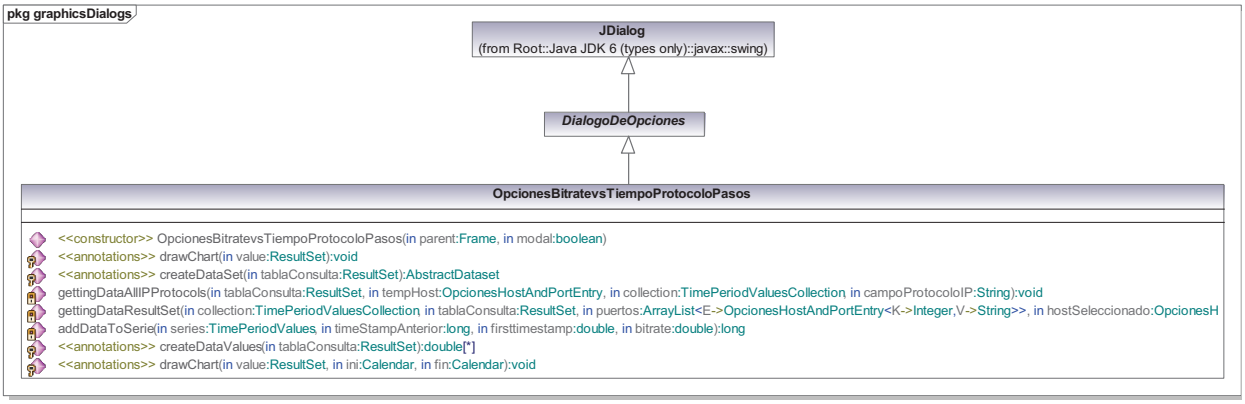


Gráfico 3.53 Clase OpcionesBitratevsTiempoProtocoloPasos

3.4.3.7.5. Clase OpcionesTimeSeriesDatasetCreator

La clase OpcionesTimeSeriesDatasetCreator es la base que implementa los métodos comunes en la obtención de los valores necesarios para la graficación de series de tiempo. Las clases que se basen en esta podrán personalizar la forma con que se presentan los datos gráficamente.

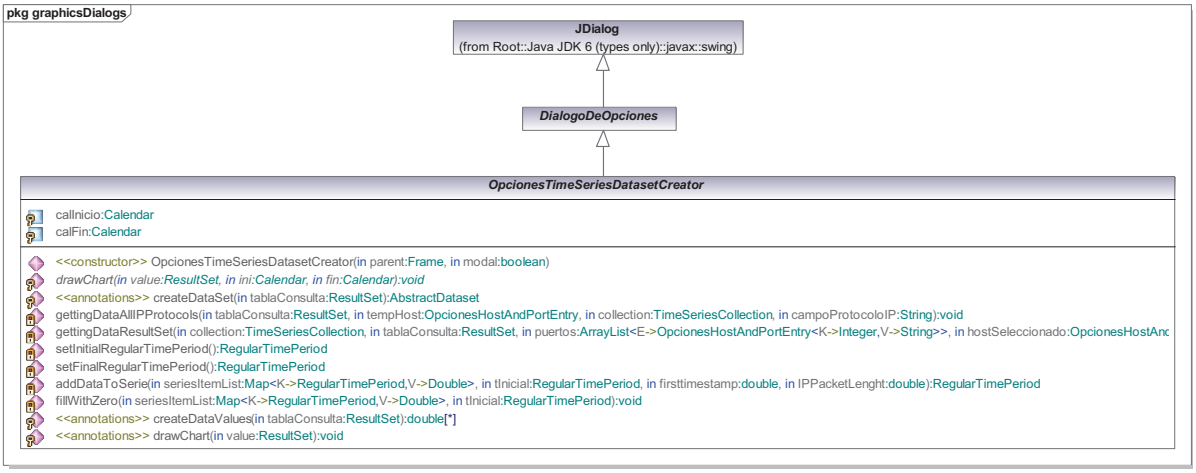


Gráfico 3.54 Clase OpcionesTimeSeriesDatasetCreator

3.4.3.7.6. Clase OpcionesSeriesTiempoProtocolos

La clase OpcionesSeriesTiempoProtocolos hereda de OpcionesTimeSeriesDatasetCreator e implementa el método de graficación, de tal manera que genere un gráfico de series de tiempo

utilizando líneas para unir los puntos correspondientes a cada intervalo regular de tiempo.

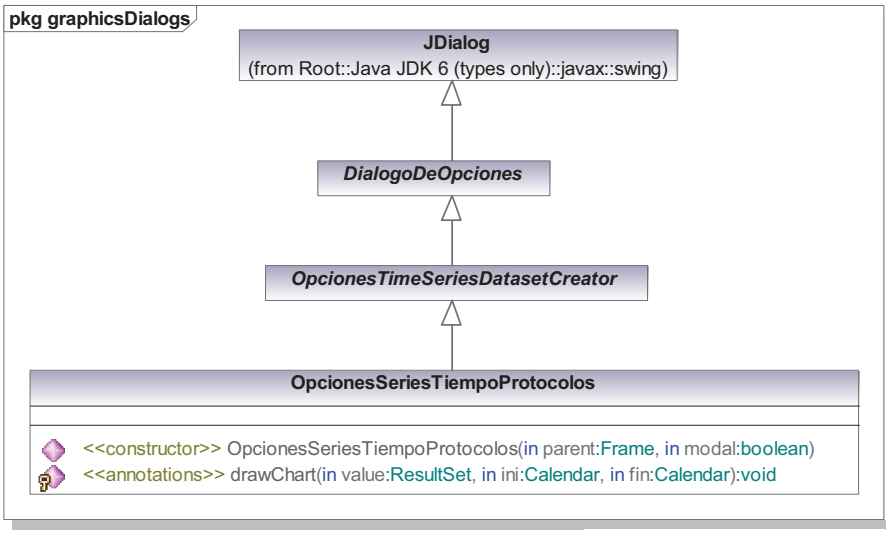


Gráfico 3.55 Clase OpcionesSeriesTiempoProtocolos

3.4.3.7.7. Clase OpcionesPasosPromedioProtocolos

La clase OpcionesPasosPromedioProtocolos hereda de OpcionesTimeSeriesDatasetCreator e implementa el método de graficación para la generación de una serie de tiempo, que dibuja pasos en lugar de líneas en cada intervalo regular de tiempo.

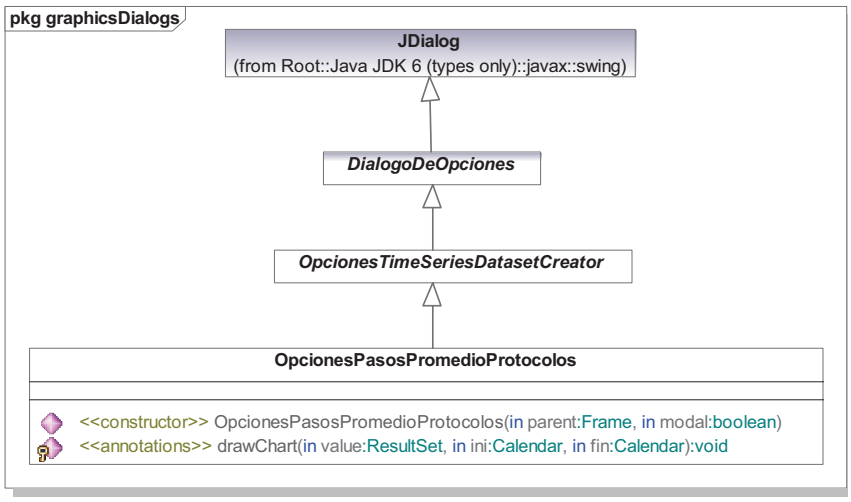


Gráfico 3.56 Clase OpcionesPasosPromedioProtocolos

3.4.3.7.8. *Clase OpcionesPorcentajeGraficoPastelBase*

La clase OpcionesPorcentajeGraficoPastelBase sirve de base para las clases de graficación de pasteles de porcentajes 2D y 3D. Al igual que todos los cuadros de diálogo obtiene los parámetros para la consulta a la base de datos por parte del usuario del programa.

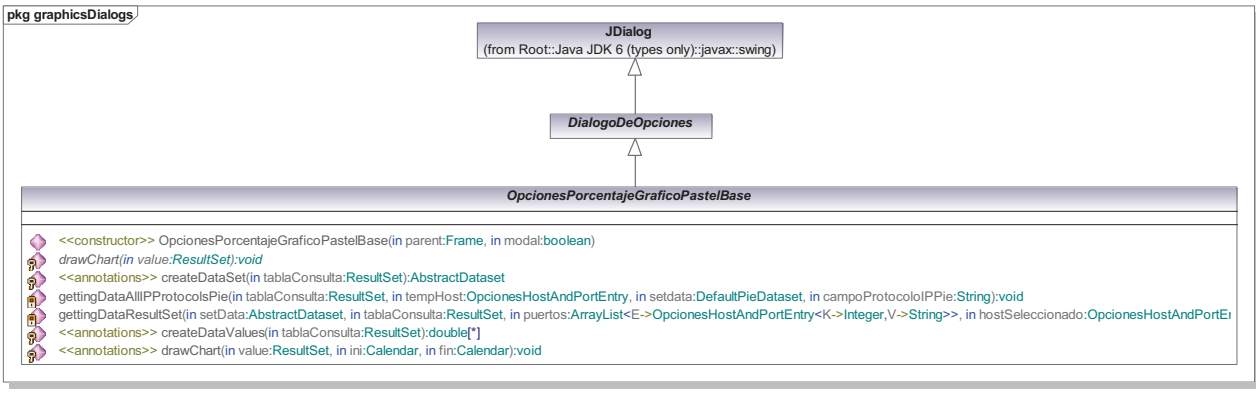


Gráfico 3.57 Clase OpcionesPorcentajeGraficoPastelBase

3.4.3.7.9. *Clase OpcionesPorcentajeGraficoPastel2D*

La clase OpcionesPorcentajeGraficoPastel2D implementa un cuadro de diálogo para obtener los parámetros necesarios en la graficación de un pastel 2D de porcentajes de datos recibidos o enviados.

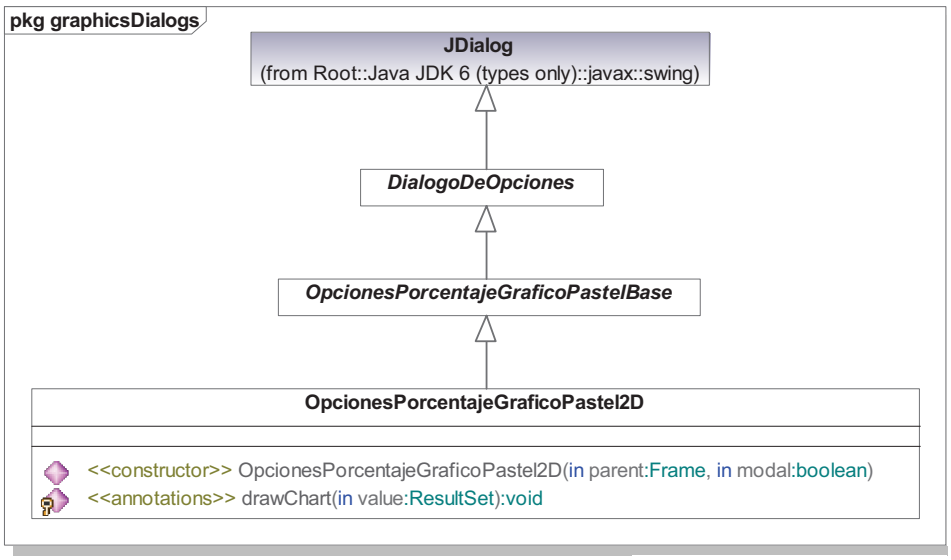


Gráfico 3.58 Clase OpcionesPorcentajeGraficoPastel2D

3.4.3.7.10. Clase OpcionesPorcentajeGraficoPastel3D

La clase OpcionesPorcentajeGraficoPastel3D realizar los mismos procedimientos que el diálogo 2D, con la excepción que el gráfico generado es de apariencia tridimensional.

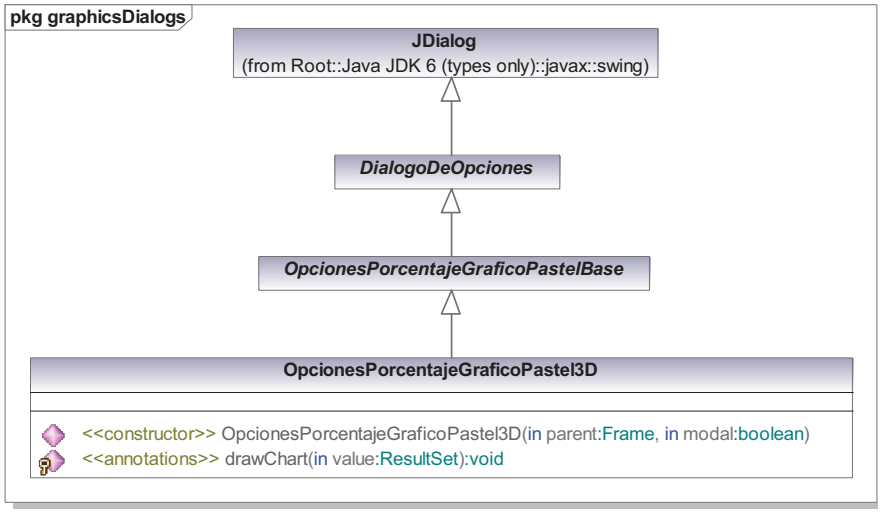


Gráfico 3.59 Clase OpcionesPorcentajeGraficoPastel3D

3.4.3.7.11. Clase OpcionesHistogramaTraficoInternet

La clase OpcionesHistogramaTraficoInternet implementa un cuadro de diálogo y recoge las opciones del usuario para la graficación de un histograma, diagrama de frecuencias acumuladas y un resumen de Estadística Descriptiva.

Gráfico 3.60 Clase OpcionesHistogramaTraficoInternet

3.4.3.7.12. Clase OpcionesGraficoBitratePromedio

La clase OpcionesGraficoBitratePromedio recoge los parámetros suministrados por el usuario y genera un diagrama de barras horizontales cuya longitud representa la tasa de trasferencia promedio del intervalo de tiempo elegido.

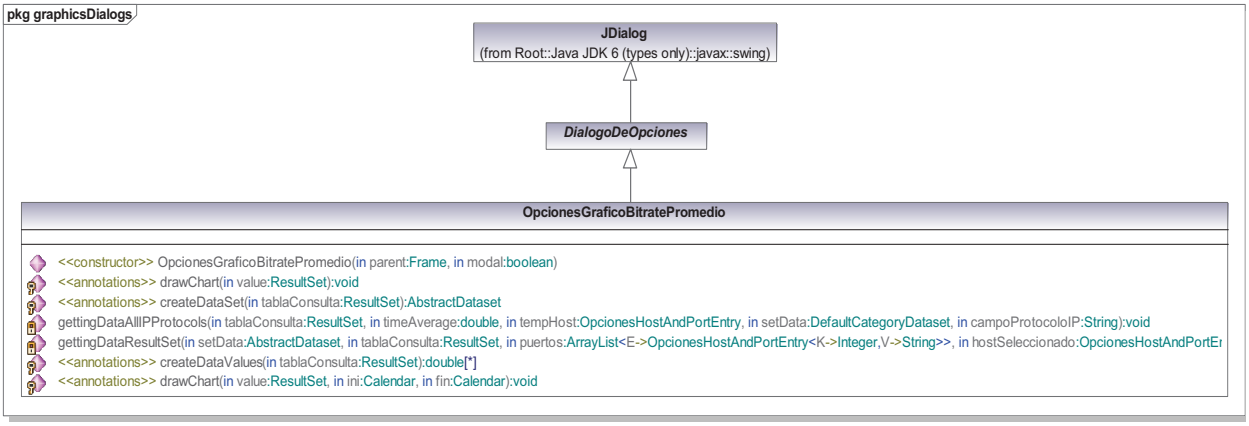


Gráfico 3.61 Clase OpcionesGraficoBitratePromedio

3.4.3.7.13. Clase OpcionesDNSReverse

La clase OpcionesDNSReverse implementa un cuadro de diálogo de selección de opciones para generar un gráfico del ranking de direcciones IP más usadas y el mecanismo de resolución de nombres.

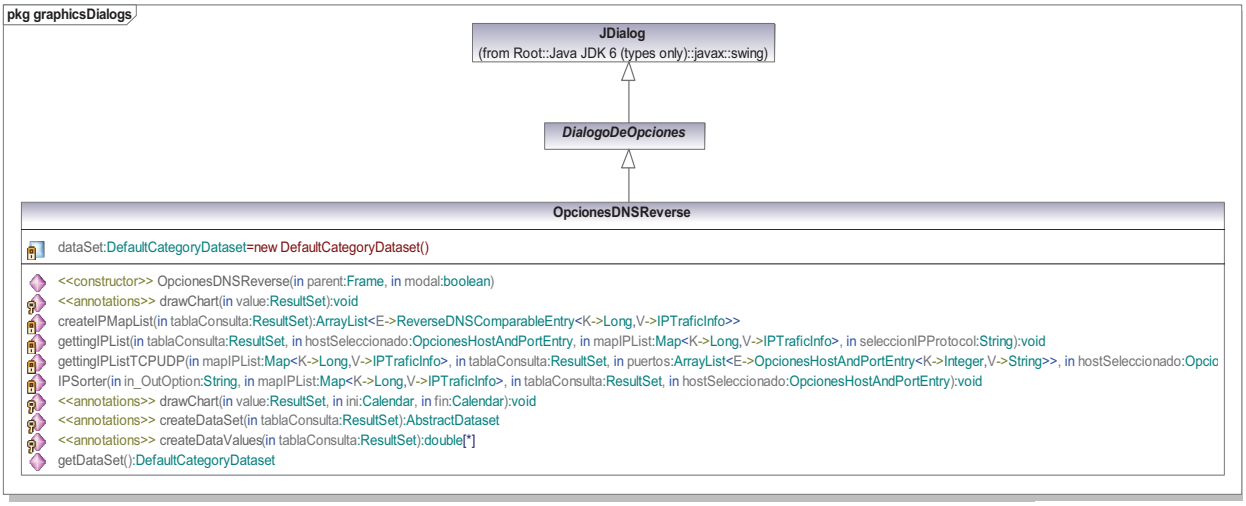


Gráfico 3.62 Clase OpcionesDNSReverse

3.4.3.8. Clases del paquete importexportdatabase

3.4.3.8.1. Clase FolderZiper

La clase FolderZiper comprime archivos y carpetas respetando su jerarquía en un archivo con extensión .zip.

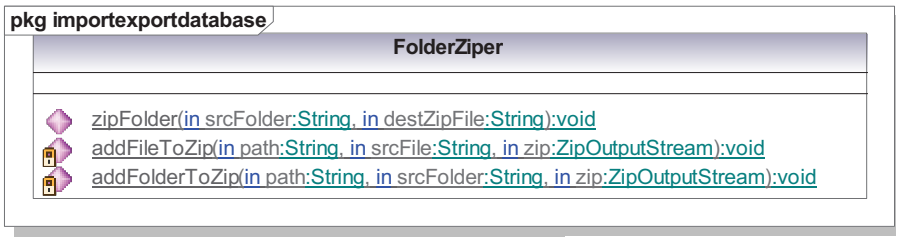


Gráfico 3.63 Clase FolderZiper

3.4.3.8.2. Clase Unzip

La clase Unzip implementa un descompresor de archivos .zip para la importación de la base de datos, además utiliza un cuadro de diálogo que permite elegir el archivo a descomprimir.

Gráfico 3.64 Clase Unzip

3.4.3.8.3. Clase ImportFileChooser

La clase ImportFileChooser permite recuperar el backup de los archivos que respaldan a la base de datos, este backup tiene la extención .zip, además dispondrá de un cuadro de diálogo para la importación.

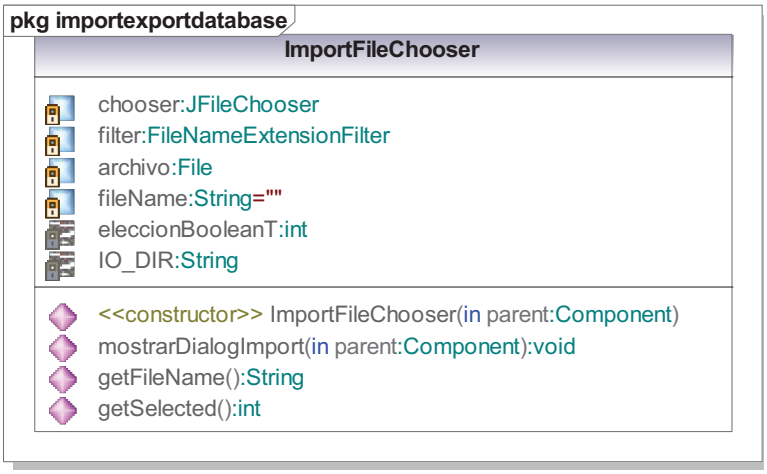


Gráfico 3.65 Clase ImportFileChooser

3.4.3.8.4. Clase ExportFileChooser

La clase ExportFileChooser permite realizar un backup de los archivos que respaldan a la base de datos, exportando todos ellos en una carpeta comprimida con la extensión .zip, dispondrá de un cuadro de diálogo para elegir el directorio y nombre del archivo.

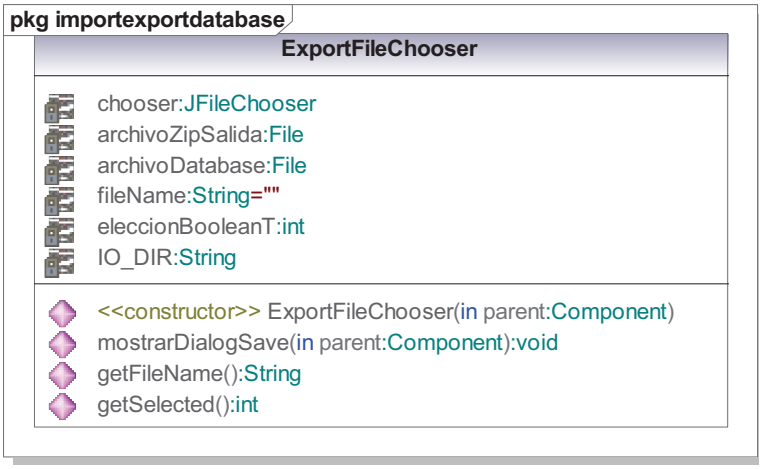


Gráfico 3.66 Clase ExportFileChooser

3.4.3.9. Clases del paquete lookandfeel

3.4.3.9.1. Clase LookAndFeelSelector

La clase LookAndFeelSelector permite instanciar un cuadro de diálogo y una lista de skins[3] contenida en un JComboBox para cambiar el tema de la interfaz gráfica.

[3] Temas visuales para personalizar la interfaz gráfica.

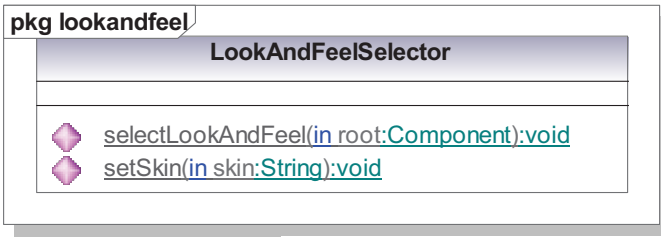


Gráfico 3.67 Clase LookAndFeelSelector

3.4.3.9.2. Clase SubstanceSkinComboSelector

La clase SubstanceSkinComboSelector carga todos los temas disponibles en un objeto JComboBox para que el usuario escoja uno de ellos y mostrarlo en la interfaz gráfica del programa.

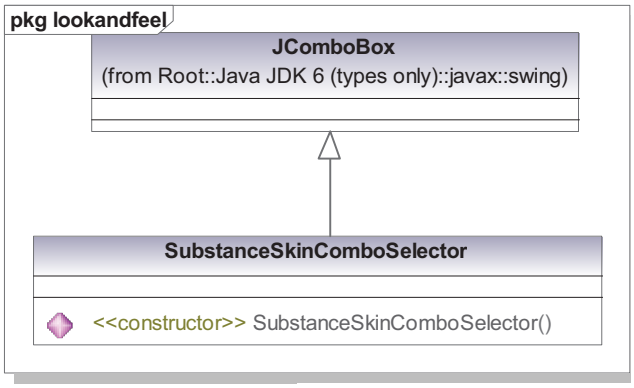


Gráfico 3.68 Clase SubstanceSkinComboSelector

3.4.3.9.3. Clase LoadLookAndFeel

La clase LoadLookAndFeel permite guardar en un archivo de texto el nombre de la plantilla visual actual, para que cuando se reinicie el programa continúe con el mismo tema de interfaz gráfica.

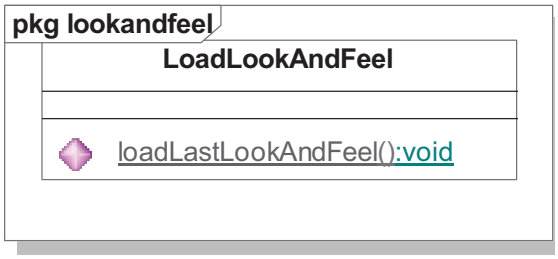


Gráfico 3.69 Clase LoadLookAndFeel

3.4.3.10. Clases del paquete minibrowser

3.4.3.10.1. Clase Browser

La clase Browser permite utilizar un navegador sea internet explorer o mozilla del sistema operativo nativo y tiene una casilla de activación que permite ocultar la barra de tareas del navegador.

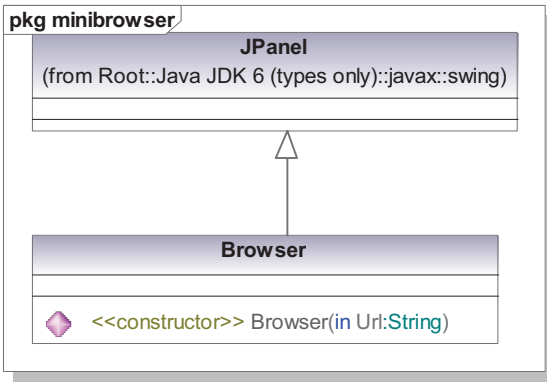


Gráfico 3.70 Clase Browser

3.4.3.11. Clases del paquete newhostdetected

3.4.3.11.1. Clase NewHostParameters

La clase NewHostParameters recupera las referencias de los diferentes parámetros necesarios para la construcción del árbol de estaciones de trabajo que se obtiene durante el monitoreo del tráfico de Internet.

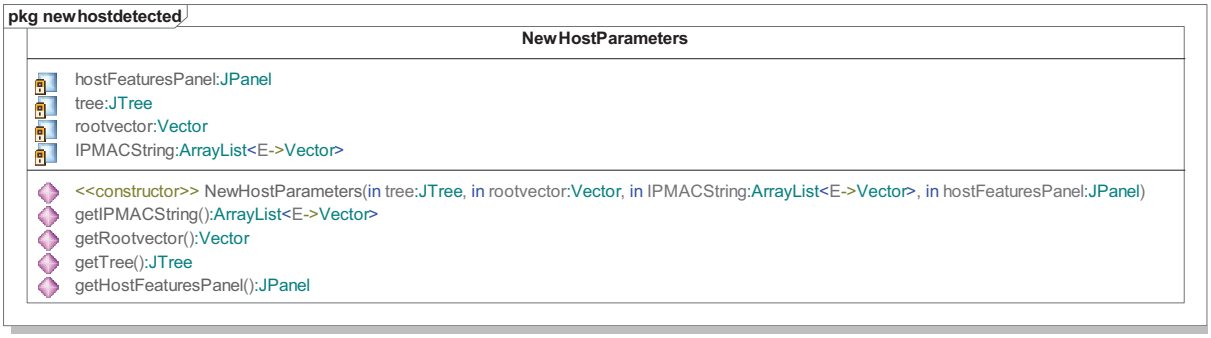


Gráfico 3.71 Clase NewHostParameters

3.4.3.11.2. Clase NewHost

La clase NewHost realiza los cambios necesarios para la creación de una estación de trabajo en un nodo del componente árbol, a la llegada de un paquete proveniente de una dirección IP nueva.

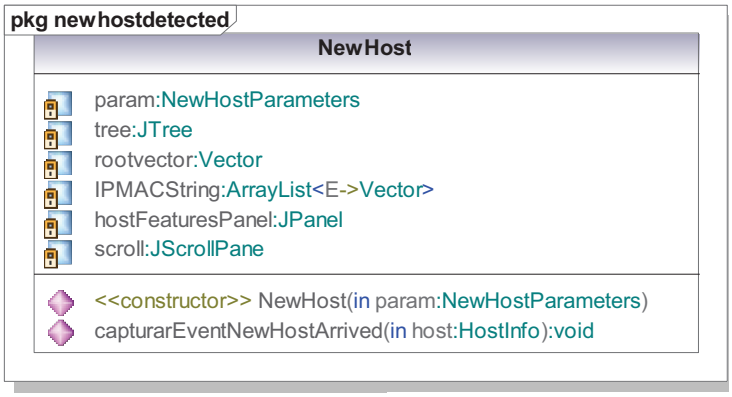


Gráfico 3.72 Clase NewHost

3.4.3.11.3. Clase NamedVectorHost

La clase NamedVectorHost crea un vector y le asigna un nombre para identificarlo, este vector se añadirá al árbol con un hijo y los objetos que contiene el vector serán las hojas del mismo.

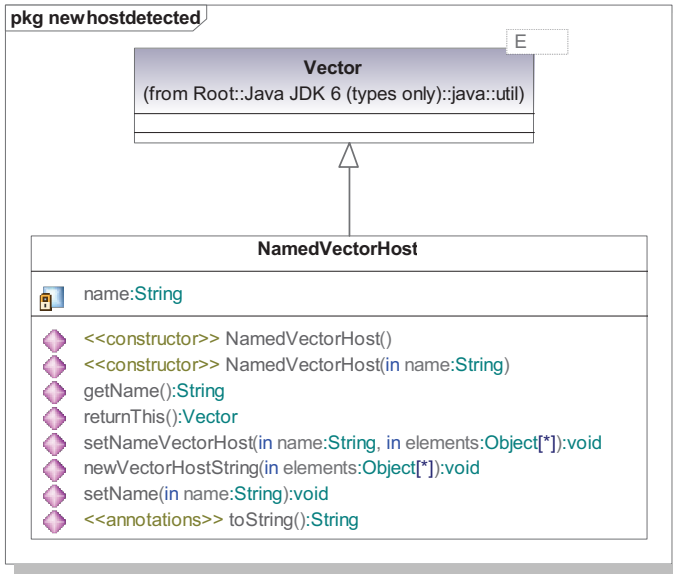


Gráfico 3.73 Clase NamedVectorHost

3.4.3.11.4. Clase CheckBoxNodeRenderer

La clase CheckBoxNodeRenderer indexa un objeto del tipo JCheckBox, como una hoja dentro de un objeto JTree, permitiendo seleccionar hosts para implementar el filtro del sniffer.

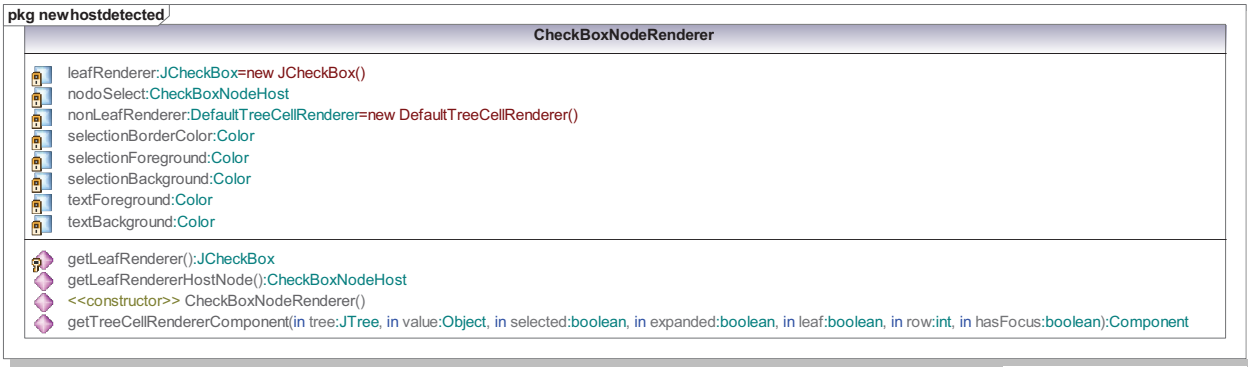


Gráfico 3.74 Clase CheckBoxNodeRenderer

3.4.3.11.5. Clase CheckBoxNodeHost

La clase CheckBoxNodeHost representa un host con su dirección IP, MAC y una bandera de selección que permite indicar al filtro si el host ha sido seleccionado.

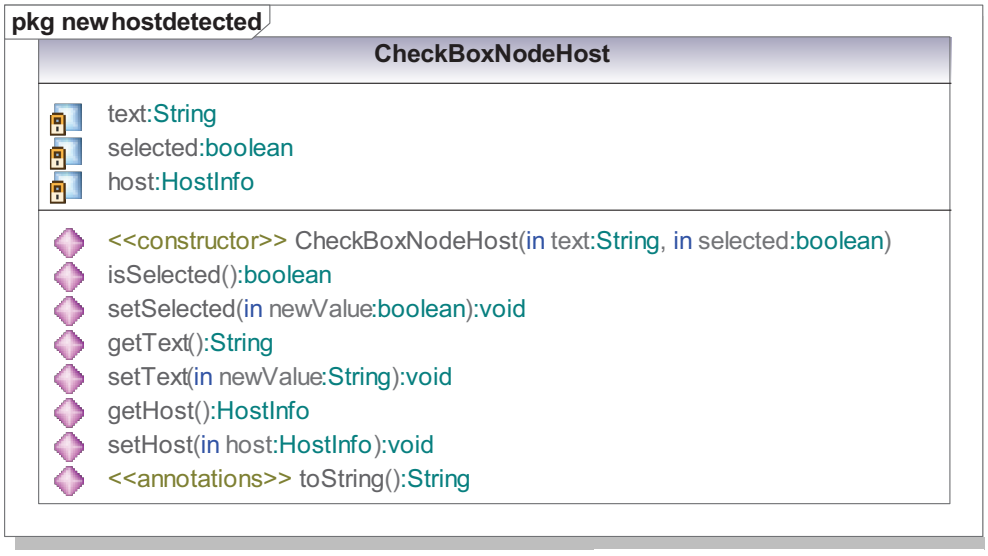


Gráfico 3.75 Clase CheckBoxNodeHost

3.4.3.11.6. Clase CheckBoxNodeEditor

La clase CheckBoxNodeEditor edita los objetos que contiene el árbol en tiempo real, cuando se realiza una acción sobre el mismo, ya sea desplegar, contraer o señalar, dependiendo de las acciones del usuario.

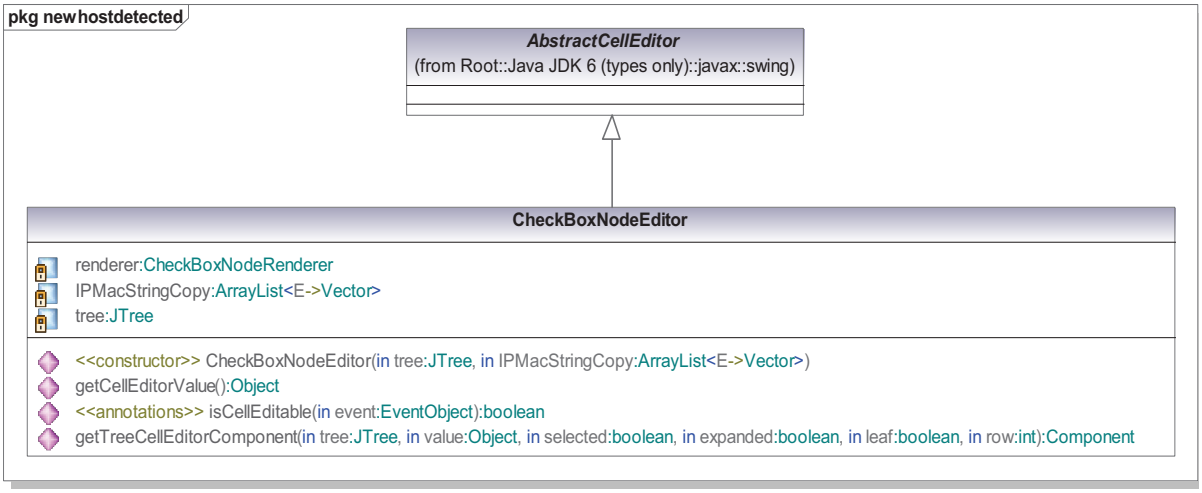


Gráfico 3.76 Clase CheckBoxNodeEditor

3.4.3.12. Clases del paquete realtimegraphs

3.4.3.12.1. Clase CustomizedPolyline

La clase CustomizedPolyline sobrescribe el método lineTo necesario para que con cada punto nuevo se forme un nuevo polígono cerrado añadiendo otro en la coordenada (x, 0).

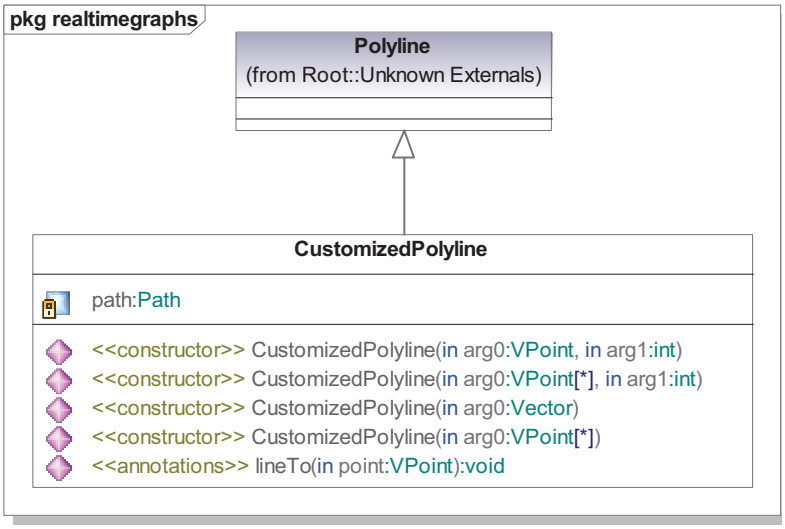


Gráfico 3.77 Clase CustomizedPolyline

3.4.3.12.2. Clase RealTimeBitrate

La clase RealTimeBitrate instancia e inicializa los objetos necesarios para la graficación de la tasa de transferencia en tiempo real, así como los atributos que permiten describir de mejor manera la representación del gráfico en un panel, por ejemplo, gradiente de color para identificar cualitativamente y a simple vista la actividad de la red, alarma de sonido cuando el límite ingresado por el usuario ha sido sobrepasado, etiquetas con el bitrate instantáneo y promedio de los últimos 15 segundos. Además incluye mecanismos para mostrar la graficación en tiempo real en el área de notificación.

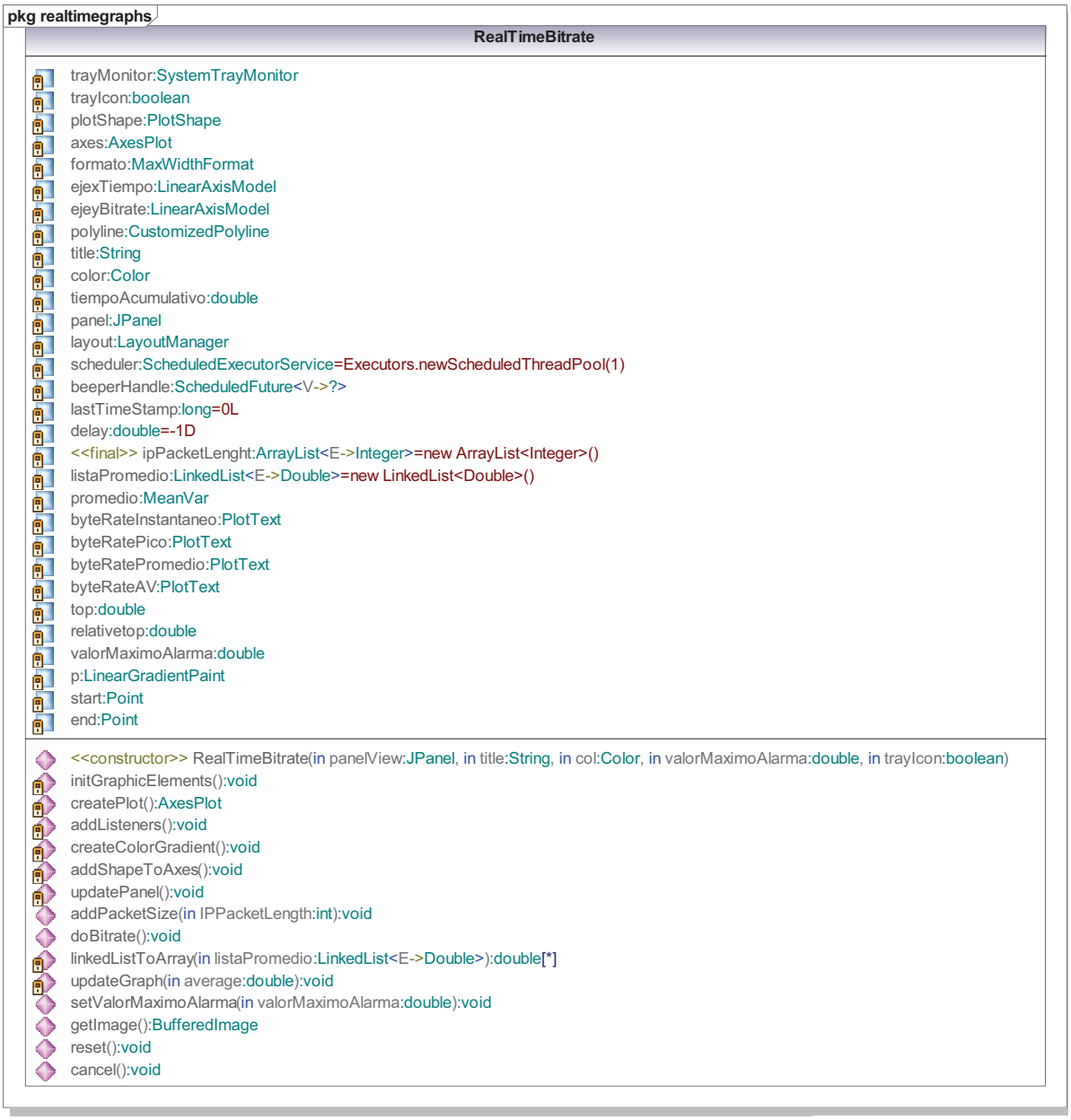


Gráfico 3.78 Clase RealTimeBitrate

La actualización del gráfico se lleva a cabo cada segundo y para que esto sea posible debe existir un objeto o método que pueda ser ejecutado de esta forma. En este caso se implementa el método run () de un objeto Runnable que cumple con las actividades detalladas en el diagrama de actividad del gráfico 3.79.

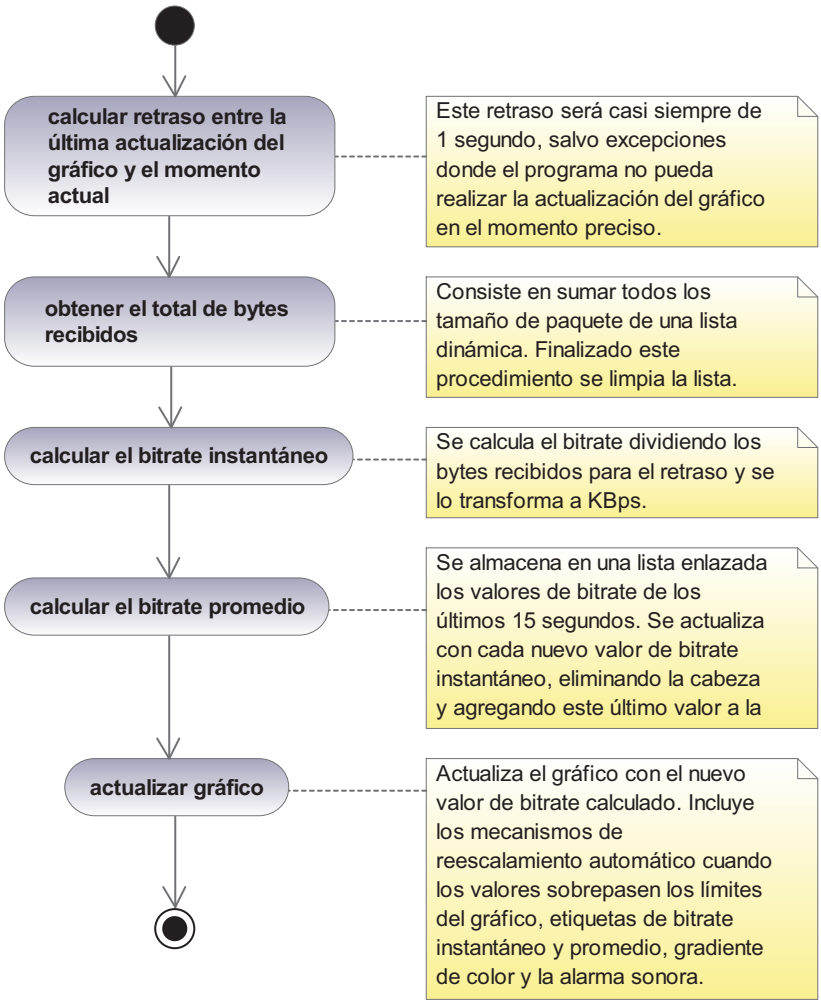


Gráfico 3.79 Diagrama de actividad del método run() del hilo de ejecución independiente de actualización del gráfico en tiempo real de la clase RealTimeBitrate

3.4.3.12.3. Clase SystemTrayMonitor

La clase SystemTrayMonitor permite añadir la representación del tráfico de Internet en tiempo real, como un ícono en el área de notificación del sistema operativo nativo, si este lo soporta.

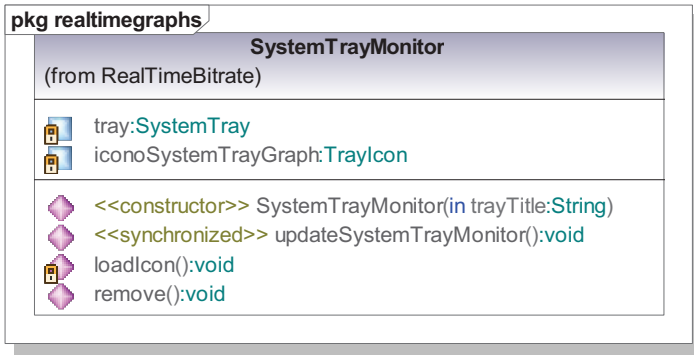


Gráfico 3.80 Clase SystemTrayMonitor

3.4.3.13. Clases del paquete reversednsresolver

3.4.3.13.1. Clase IPTraficInfo

La clase IPTraficInfo contiene información del tipo de protocolo, puerto, cantidad de bytes y lista de direcciones para una posterior resolución de nombres. Además implementa una función de comparación de objetos que servirá para ordenarlos descendentemente según el valor de bytes enviados o recibidos y sobrescribe el método toString() para que retorne la representación en una cadena de caracteres del contenido de los atributos de la clase.

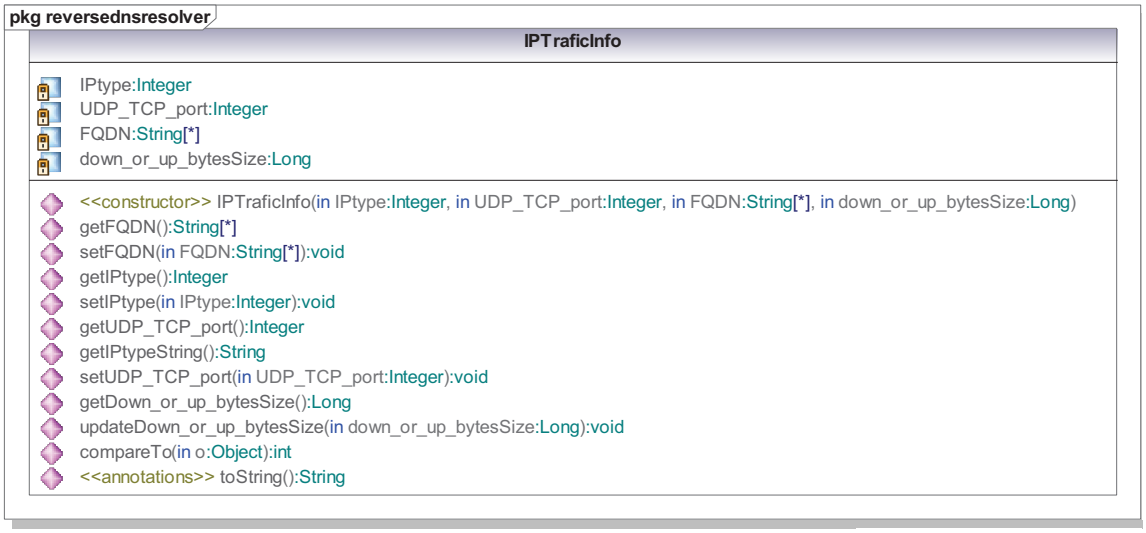


Gráfico 3.81 Clase IPTraficInfo

3.4.3.13.2. Clase ReverseDNSComparableEntry

La clase ReverseDNSComparableEntry es un objeto que asocia una dirección IP con un objeto IPTraficInfo e implementa la funcionalidad de comparación, para poder ordenarlos en una lista dinámica posteriormente.

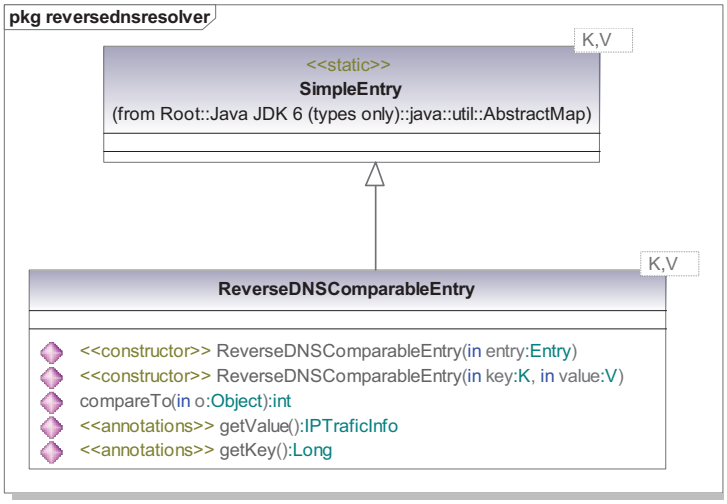


Gráfico 3.82 Clase ReverseDNSComparableEntry

3.4.3.13.3. Clase ReverseIPListResolution

La clase ReverseIPListResolution realiza la resolución de nombres de dominio para las diferentes direcciones IPs contenidas dentro de una lista con su respectiva descripción.

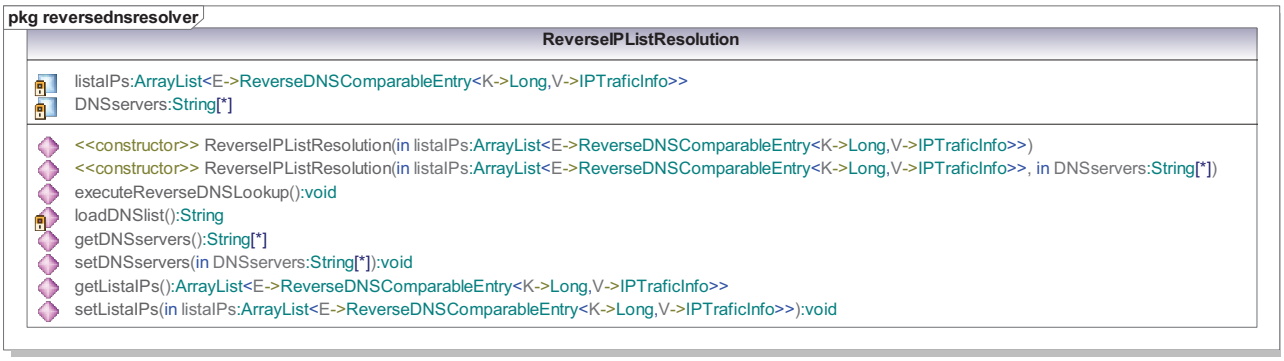


Gráfico 3.83 Clase ReverseIPListResolution

3.4.3.14. Clases del paquete save

3.4.3.14.1. Clase AutoSaveOptionPane

La clase AutoSaveOptionPane crea una ventana que muestra opciones, para que el usuario guarde una cadena de caracteres del área de texto en la cual se despliega los paquetes decodificados de la captura, en caso de no elegir una de las opciones presentadas en el cuadro de diálogo, la ventana se cerrará automáticamente después de 5 segundos.

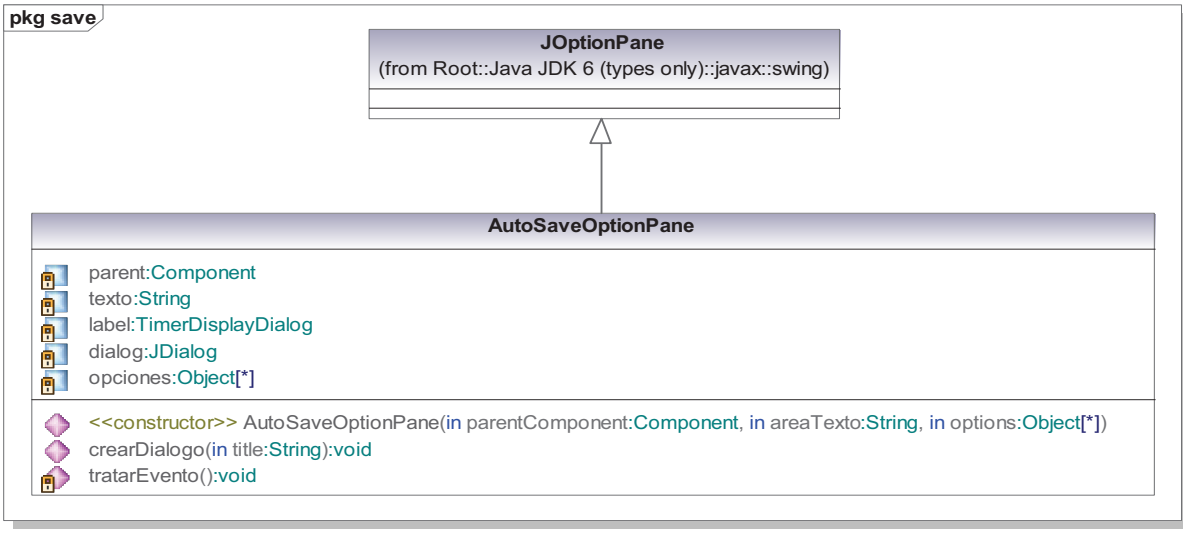


Gráfico 3.84 Clase AutoSaveOptionPane

3.4.3.14.2. Clase TimerDisplayDialog

La clase TimerDisplayDialog es una etiqueta un contador regresivo que lanza un evento para cerrar el cuadro de diálogo.

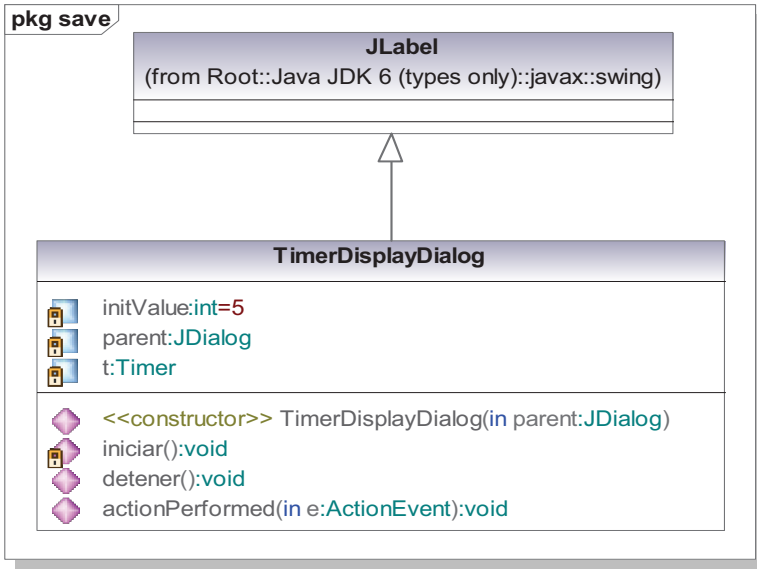


Gráfico 3.85 Clase TimerDisplayDialog

3.4.3.14.3. Clase FileUtils

La clase FileUtils contiene métodos para escribir o borrar los datos obtenidos de un área de texto o guardar una imagen en un archivo y de esta manera respaldar la información del usuario.

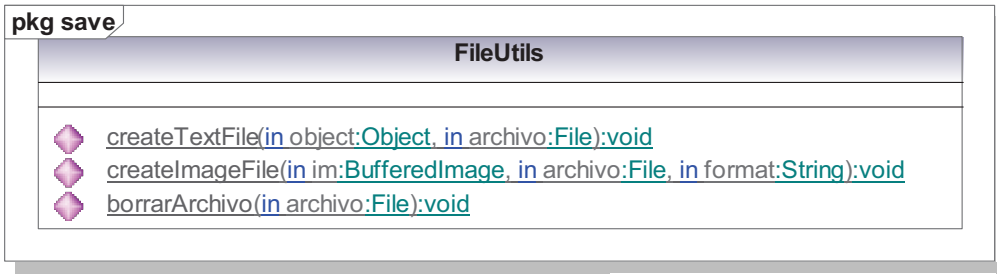


Gráfico 3.86 Clase FileUtils

3.4.3.14.4. Clase AutoSave

La clase AutoSave implementa la funcionalidad de auto-guardado de un archivo de texto que contiene una cadena de caracteres de los paquetes capturados y mostrados en un área de texto.

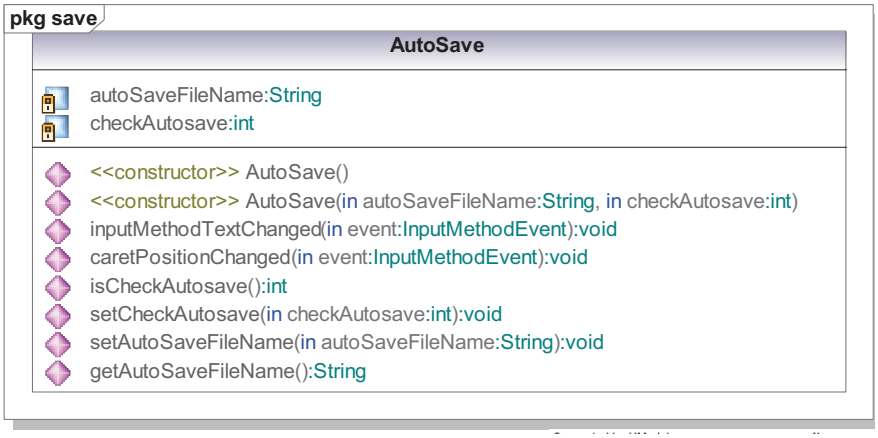


Gráfico 3.87 Clase AutoSave

3.4.3.14.5. Clase SaveTextFileChooser

La clase SaveTextFileChooser permite guardar un archivo con extensión .txt de los datos obtenidos de un área de texto, con este objetivo se despliega un cuadro de diálogo para elegir el directorio y nombre del archivo.

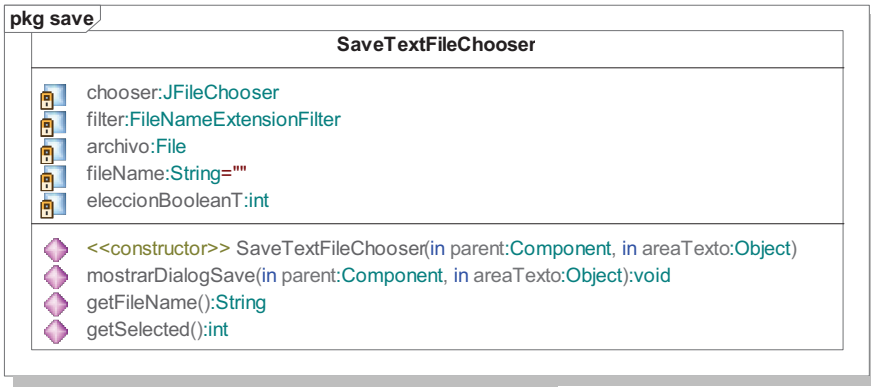


Gráfico 3.88 Clase SaveTextFileChooser

3.4.3.14.6. Clase SaveImageFileChooser

La clase SaveImageFileChooser permite guardar un archivo de imagen (con extensión *.jpeg,*.png o *.bmp) de los datos obtenidos de un panel con el gráfico en tiempo real del tráfico monitoreado, del histograma de frecuencias y el polígono de frecuencias acumuladas en el análisis estadístico. Para ello se despliega un cuadro de diálogo de selección de directorio y nombre del archivo.

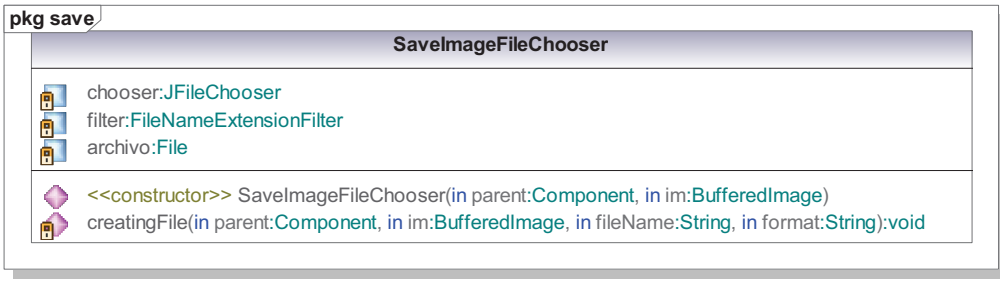


Gráfico 3.89 Clase SaveImageFileChooser

3.4.3.15. Clases del paquete sniffer

3.4.3.15.1. Clase SnifferParameters

La clase SnifferParameters contiene un conjunto de atributos, que identifican las opciones seleccionadas por el usuario a manera de filtro, para la funcionalidad del sniffer y la graficación en tiempo real del tráfico de Internet para las estaciones de trabajo seleccionadas.

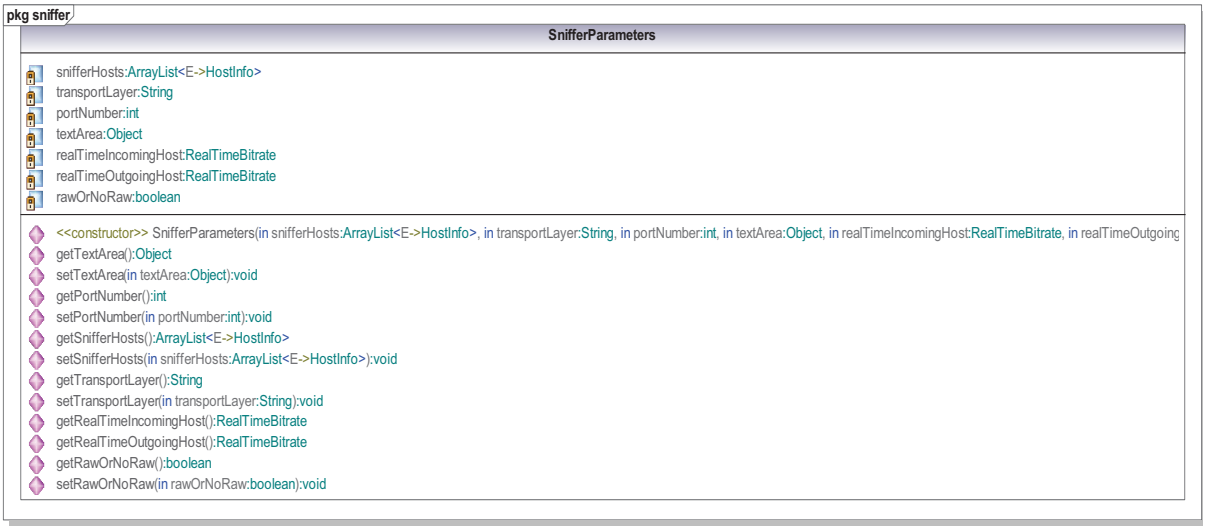


Gráfico 3.90 Clase SnifferParameters

3.4.3.15.2. Clase FilterAction

La clase FilterAction incluye un método para recuperar de la interfaz gráfica las opciones elegidas por el usuario, retornando un objeto SnifferParameters.

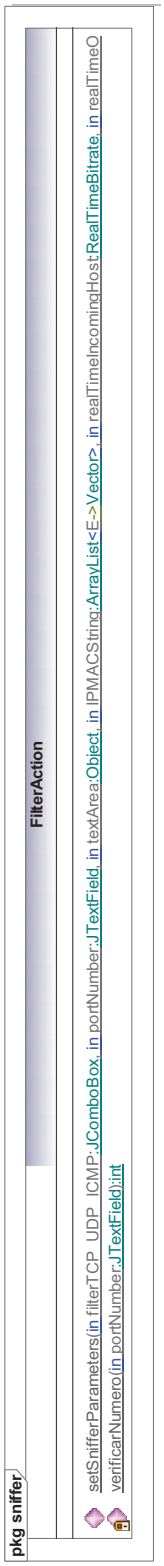


Gráfico 3.91 Clase FilterAction

3.4.3.15.3. Clase SnifferRunnable

La clase SnifferRunnable implementa el método run() que será ejecutado por un hilo independiente. Usa las opciones de filtrado seleccionadas por el usuario y genera una cadena de caracteres que representa el paquete decodificado (decodificación total o solo de cabeceras de los protocolos), esto es visualizado en un área de texto de la interfaz visual. La información de los paquetes filtrados también es utilizada para la graficación en tiempo real de la tasa de transferencia de tráfico entrante y saliente, según las opciones del filtro.

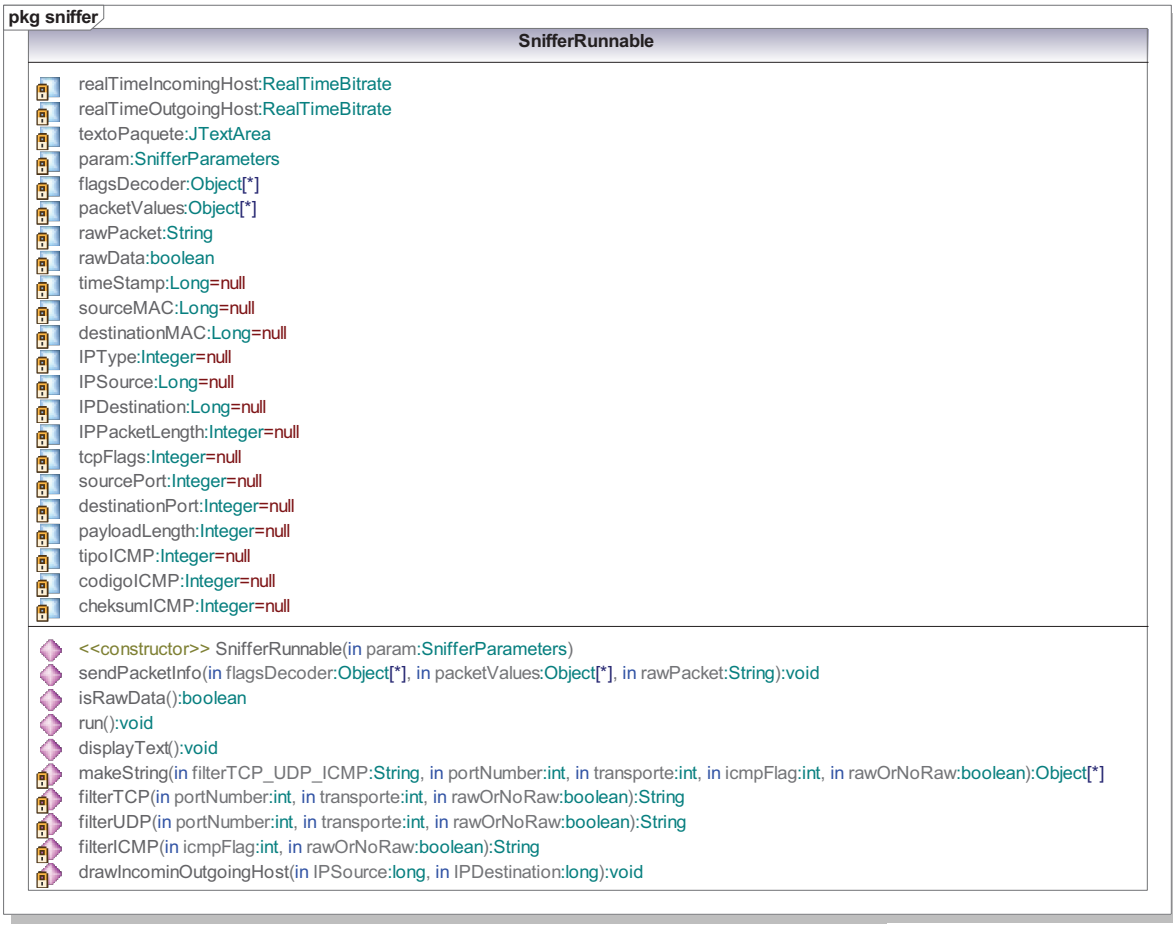


Gráfico 3.92 Clase SnifferRunnable

El gráfico 3.93 se detalla el flujo de actividades que SnifferRunnable realiza cuando se ejecuta el método run() implementado.

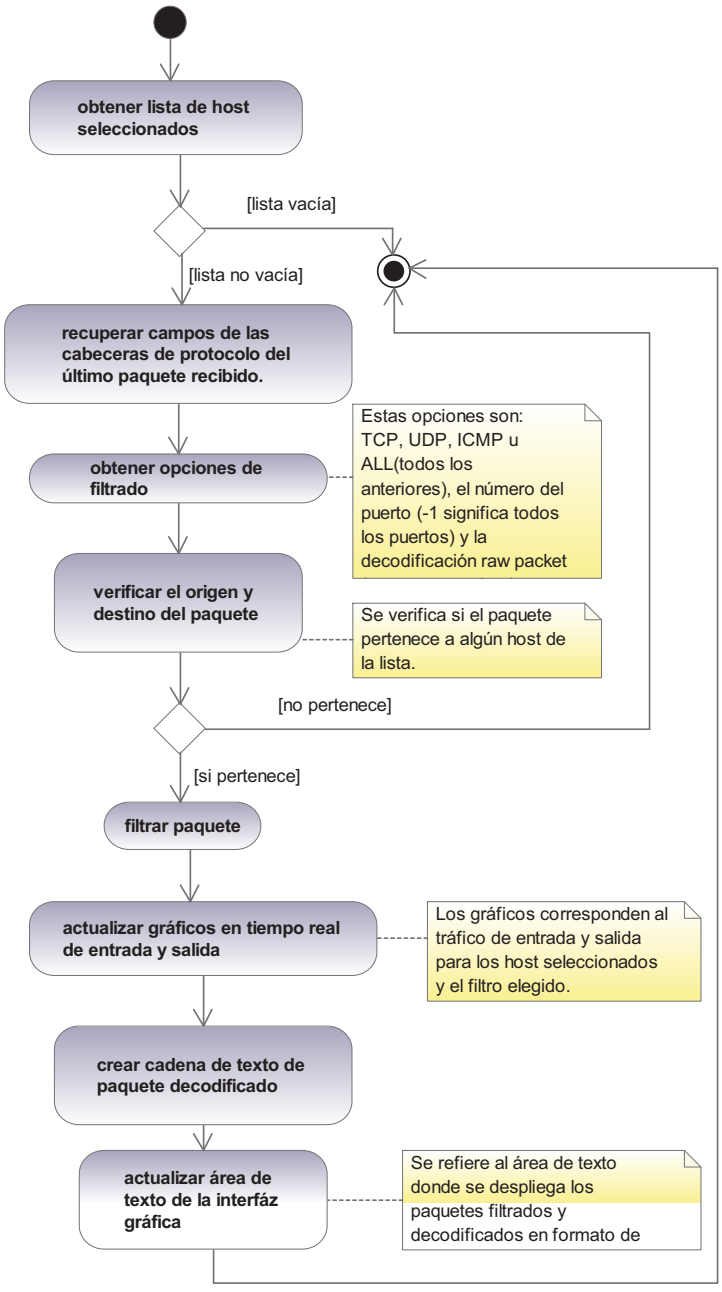


Gráfico 3.93 Diagrama de actividad del método run() de la clase SnifferRunnable

3.4.3.15.4. Clase SnifferThread

La clase SnifferThread crea un hilo que permite ejecutar el método run() de la clase SnifferRunnable. El método antes citado se debe ejecutar cada vez que se capture un nuevo paquete en el dispositivo de red.

Gráfico 3.94 Clase SnifferThread

3.4.3.16. Clases del paquete statisticalgraphics

Todas las clases de este paquete que heredan de JDialog incluyen la característica de auto dimensionamiento de acuerdo al tamaño del monitor de la PC.

Estas clases generan los gráficos y resúmenes informativos del tráfico de Internet, en trabajo conjunto con las clases del paquete graphicsDialogs (recuperan datos y los organizan de acuerdo a las opciones ingresadas por el usuario del programa). Cada una de ellas contiene un botón que abre una ventana para mostrar el contenido de ayuda e interpretación de resultados.

Cada gráfico generado proporciona una ventana emergente, que aparece al realizar un clic derecho sobre la imagen. Incluye opciones para copiar la imagen al portapapeles, grabar la imagen en formato png, herramientas de zoom, autoescala y personalización de las propiedades del gráfico como modificar el título, las etiquetas de los ejes, color de bordes, entre otros.

3.4.3.16.1. Clase HistogramaAndCumulativeFrequencyPolygon

Una instancia de esta clase genera un histograma de frecuencias y su representación tabulada, el correspondiente polígono de frecuencias acumuladas y una tabla con un resumen básico de Estadística Descriptiva.

Todos estos cálculos se los hace a partir de un array de valores de tasa de transferencia (la clase OpcionesHistogramaTraficoInternet provee de estos datos) que deben ser ingresados como argumento. Incluye acceso a una ventana de ayuda y sugerencias, opciones para redibujar los diagramas y recalcular los

valores variando el número de intervalos del rango de tasa de transferencia de datos, según se requiera.

La implementación de los gráficos y las tablas se hacen de forma modular instanciando las clases Histograma, CumulativeFrequencyPolygon, TablaDeFrecuenciasPanel y TablaDeEstadisticaDescriptiva.

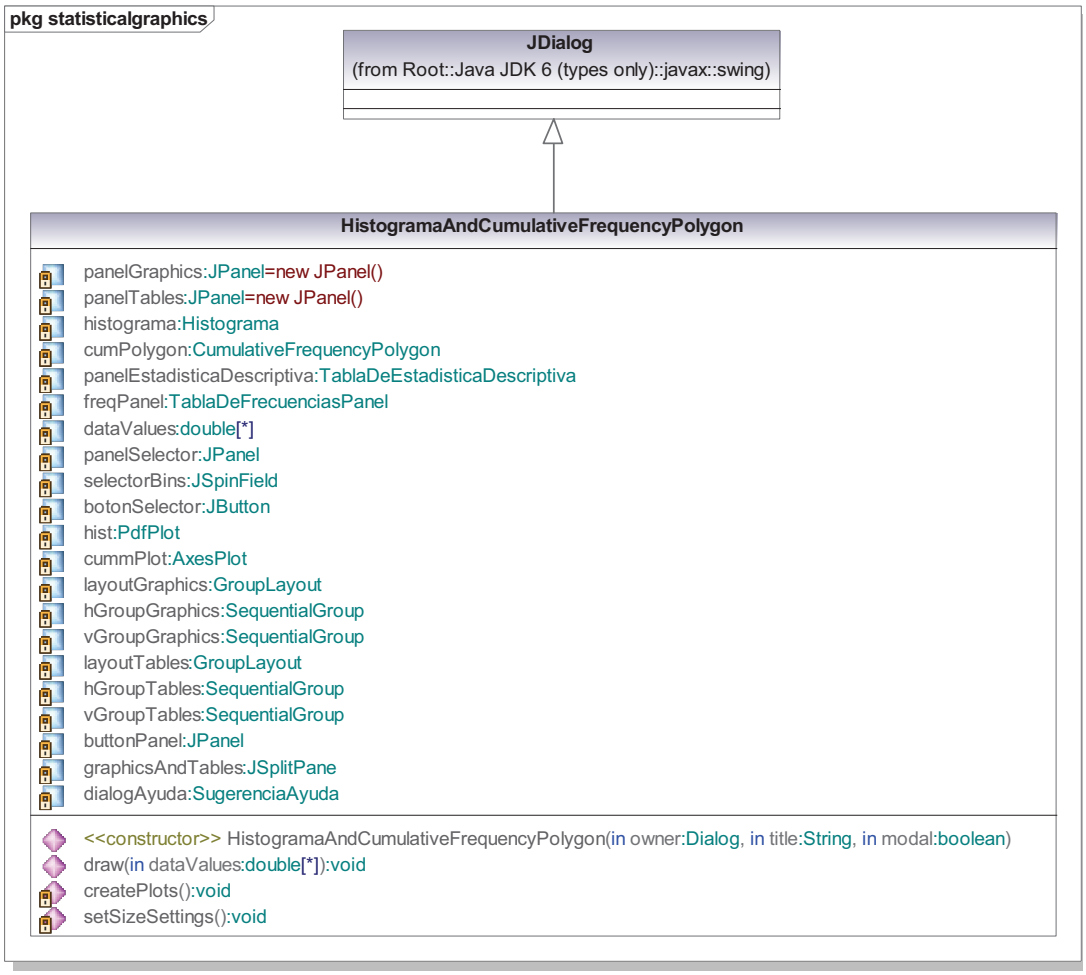


Gráfico 3.95 Clase HistogramaAndCumulativeFrequencyPolygon

3.4.3.16.2. Clase Histograma

La clase Histograma implementa un conjunto de métodos para devolver un componente gráfico que contiene un histograma de frecuencias, generado a partir de un objeto FrequencyTable[4] ingresado como parámetro.

[4] La clase FrequencyTable del paquete estadístico JSC 1.0 representa una tabla de frecuencias y provee de métodos que retornan todos los parámetros necesarios para reconstruirla gráficamente.

Un histograma es una gráfica que permite observar, en que regiones de un rango de valores, existe una mayor concentración de datos y en donde su presencia es menor. De esta forma se puede tener una idea general del comportamiento de la red con respecto a la tasa de transferencia. Como complemento se incluye en el diagrama el polígono de frecuencias correspondiente.

Para mejorar la apariencia visual del gráfico se han personalizado todos los elementos posibles, por ejemplo, el tipo, el tamaño, el color y el formato de palabras, ejes de referencia y líneas. Para utilizar la funcionalidad de un menú emergente de guardado e impresión de la imagen utiliza la clase CustomizedPdfPlot.

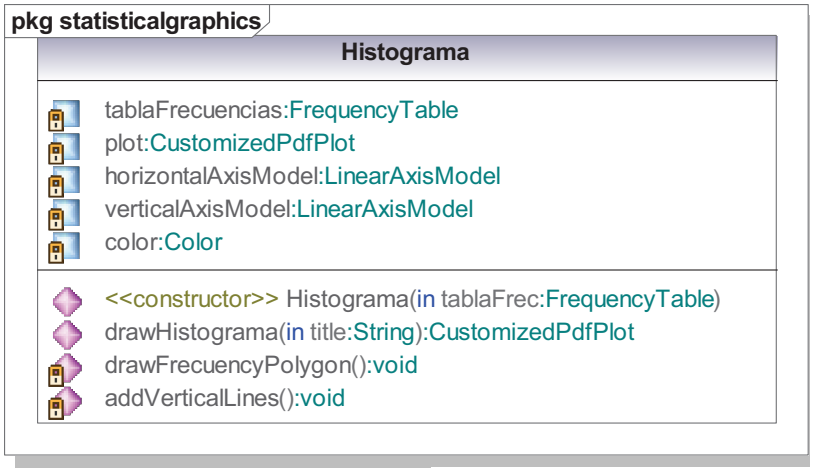


Gráfico 3.96 Clase Histograma

3.4.3.16.3. Clase CustomizedPdfPlot

La clase CustomizedPdfPlot representa un gráfico de las funciones de densidad de probabilidad creado a partir de una tabla de frecuencias. Se implementa un constructor muy completo para personalizar todos los elementos del gráfico. También se añade un menú emergente para guardar e imprimir la imagen, que se activa al hacer clic derecho sobre esta.

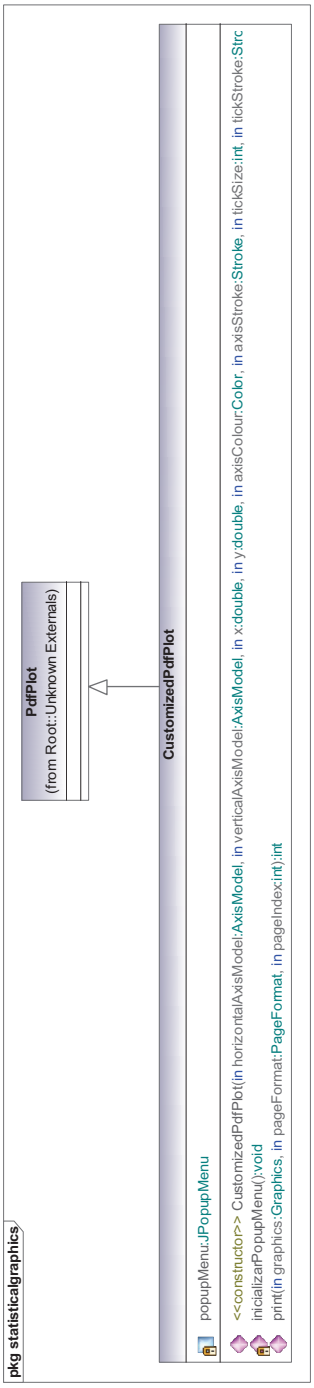


Gráfico 3.97 Clase CustomizedPdfPlot

3.4.3.16.4. Clase MousePopupListener de CustomizedPdfPlot

La clase interna MousePopupListener permite atender los eventos generados por el mouse dentro del área que contiene el gráfico. Solo muestra el menú emergente cuando se ha producido el evento clic derecho. No puede ser implementada externamente como una clase independiente porque solo maneja el objeto JPopupMenu de su clase contenedora.

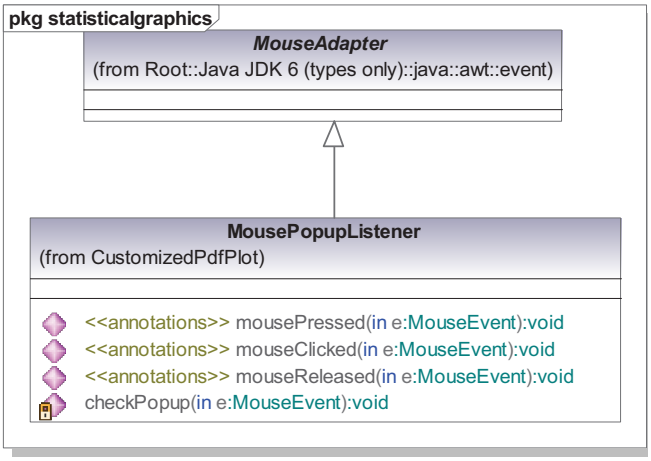


Gráfico 3.98 Clase MousePopupListener de CustomizedPdfPlot

3.4.3.16.5. Clase CumulativeFrequencyPolygon

La clase CumulativeFrequencyPolygon representa un polígono de frecuencias acumuladas en porcentajes, generado a partir de los datos de un objeto FrequencyTable.

Esta implementación personaliza las características de color, tamaño y formato de ejes de referencia, números y etiquetas de título. Además incluye la funcionalidad de guardado e impresión de la imagen mediante un menú emergente de la clase interna CustomizedAxesPlot.

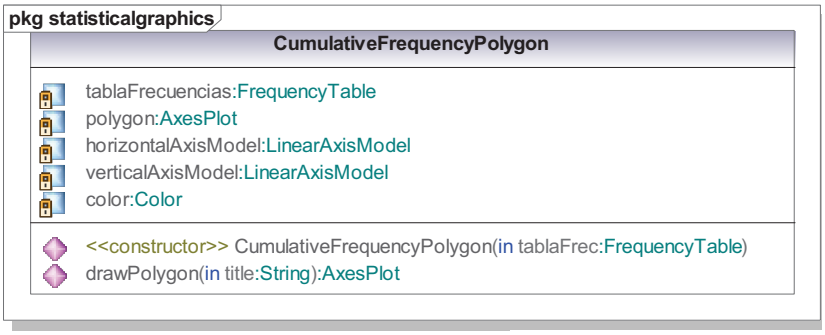


Gráfico 3.99 Clase CumulativeFrequencyPolygon

3.4.3.16.6. Clase CustomizedAxesPlot

La clase interna CustomizedAxesPlot crea un componente gráfico bidimensional que consiste de un título y dos ejes, de uso exclusivo de CumulativeFrequencyPolygon. Tiene la particularidad de mostrar un menú emergente al recibir un evento de clic derecho, para guardar e imprimir todos los objetos gráficos que hayan sido dibujados en un objeto de esta clase como base.

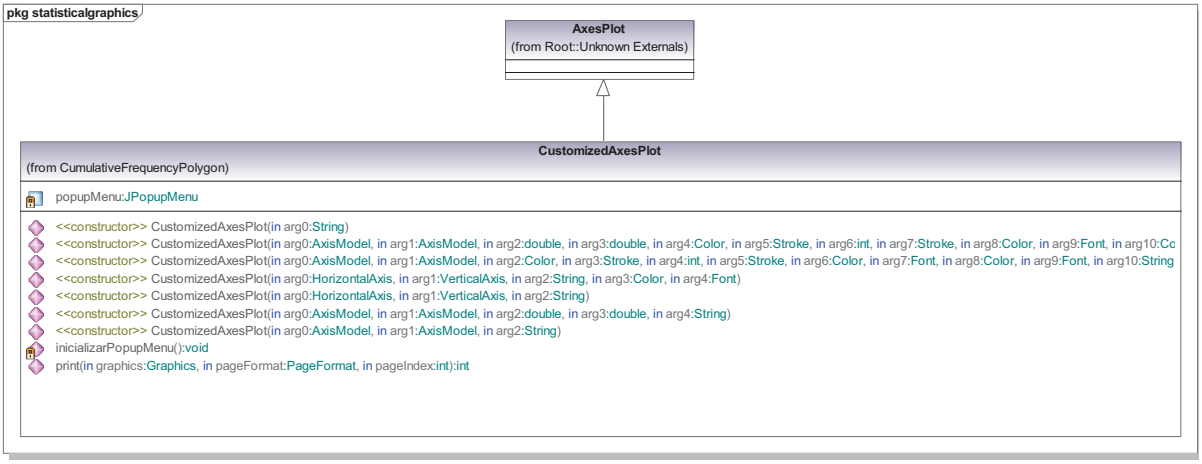


Gráfico 3.100 Clase CustomizedAxesPlot

3.4.3.16.7. Clase MousePopupListener de CustomizedAxesPlot

Implementa la misma funcionalidad de la clase MousePopupListener de CustomizedPdfPlot, pero para una clase contenedora distinta.

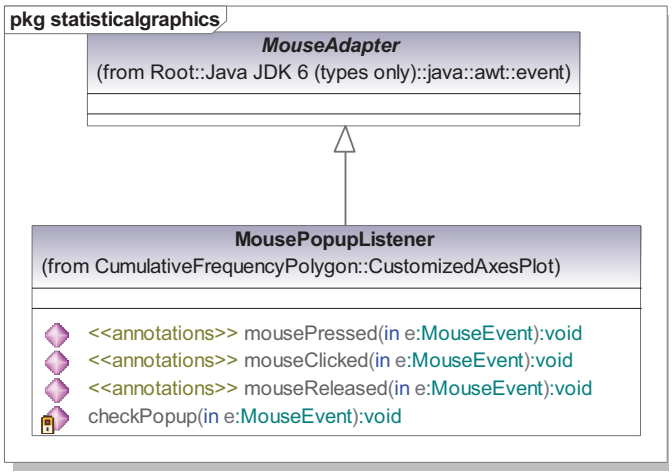


Gráfico 3.101 Clase MousePopupListener de CustomizedAxesPlot

3.4.3.16.8. Clase TablaDeFrecuenciasPanel

La clase TablaDeFrecuenciasPanel representa una representación gráfica tabulada de una tabla de frecuencias con la información procedente de un objeto FrequencyTable. El contenido puede ser copiado al portapapeles seleccionado los campos y filas y presionando la tecla de la letra "C".

La tabla de frecuencias contiene la misma información mostrada en el gráfico del histograma, pero en forma tabulada, para acceder a los valores cuantitativos y obtener una apreciación más detallada.

La tabla contiene los siguientes campos:

- Intervalos de Clase [KBps]: Representa los rangos de separación uniforme para los distintos valores de bitrate.
- Frecuencia: El número de datos que existe en cada Intervalo de Clase
- Frecuencia Relativa: La proporción obtenida de dividir el valor de frecuencia para el total de datos de la muestra.
- Frecuencia Relativa Acumulada: Corresponde a la suma de las frecuencias relativas anteriores.
- Densidad: Representa la frecuencia relativa dividida entre el ancho de clase. Este cálculo es especialmente útil cuando se tienen intervalos de clase no uniformes, pues las clases anchas tienden a contener más datos que las clases más angostas. Se la calcula para corregir el efecto de la tendencia y que el ancho de la clase no influya en la apreciación de real de la concentración de datos, en un determinado rango o intervalo e clase.

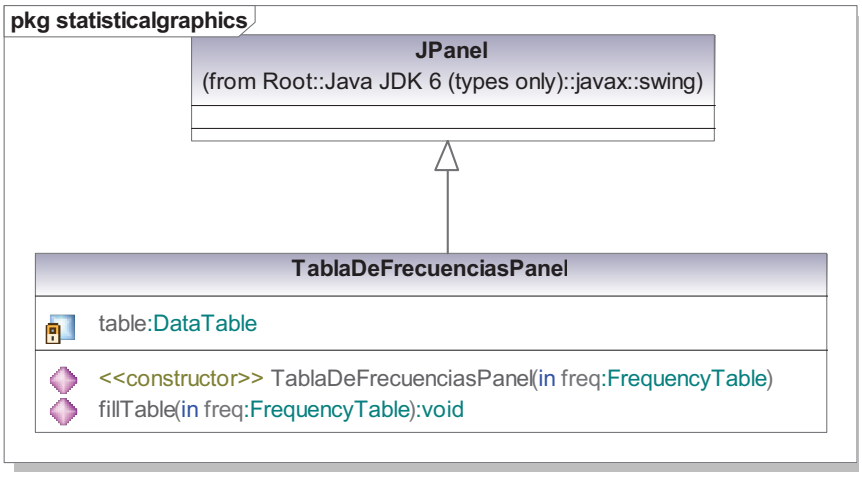


Gráfico 3.102 Clase TablaDeFrecuenciasPanel

3.4.3.16.9. Clase TablaDeEstadisticaDescriptiva

La clase TablaDeEstadisticaDescriptiva permite crear dos tablas que contendrán valores estadísticos de la media, valores máximos, mínimos, cuartiles, mediana, la media recortada y error estándar de la media, obtenidos a partir de un array de bitrates y añadidos a un panel con los histogramas correspondientes.

A continuación se explica brevemente cada columna:

- Variable: Nombre de la variable, en este caso 'Bitrate'.
- N: Tamaño de la muestra, representa el número de valores de bitrate utilizados en el cálculo.
- Media: Valor de la media muestral.
- Error estándar de la media
- Media recortada al 5%: Valor de la media sin tomar en cuenta el 5% de los datos más bajos y el 5% de los datos más altos.
- Desviación Estándar: es una cantidad que mide el grado de dispersión de una muestra.
- Valor Mínimo: El valor más pequeño de la muestra.

Los cuartiles son valores que dividen al conjunto de datos en cuatro partes iguales.

- Q1: Primer cuartil.
- Mediana: representa una medida de tendencia central de los datos obtenidos en una muestra. Coincide con el segundo cuartil Q2.
- Q3: Tercer cuartil.
- Valor Máximo: El mayor valor de la muestra

Los campos más importantes que el usuario debe tomar en cuenta son sin duda, la media y la desviación estándar. El primero indica el valor promedio de bitrate en el rango de tiempo seleccionado y el segundo representa el grado de dispersión de los valores de bitrate, este último mientras más cercano a cero sea su valor, menor será la desviación con respecto a la media, es decir, que el tráfico se mantuvo casi constante a lo largo del tiempo, caso contrario si el valor de bitrate cambió con el tiempo, implica un aumento en el valor de la desviación estándar.

Los valores Q1, mediana, Q2 y la media recortada son útiles pues no están muy afectados por valores mucho más grandes (representados por el valor máximo) o pequeños (representados por el valor mínimo) que el resto, estos datos considerados atípicos podrían tratarse de valores picos de la conexión y no corresponden al valores reales de tráfico de Internet.

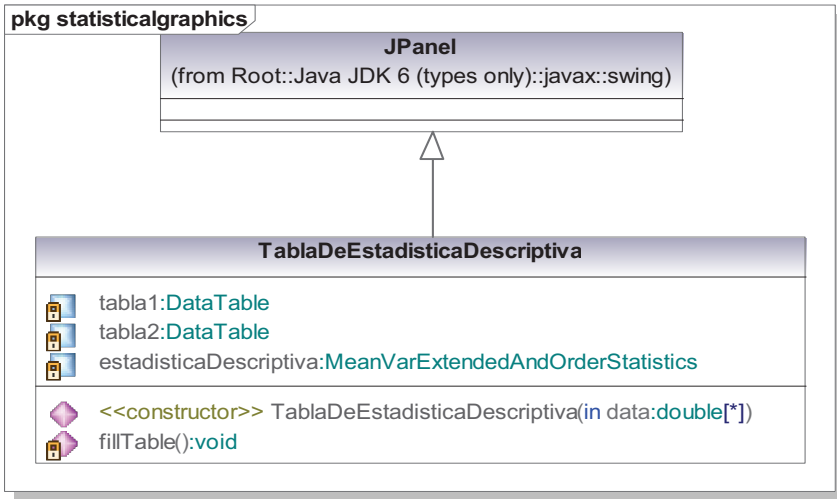


Gráfico 3.103 Clase TablaDeEstadisticaDescriptiva

3.4.3.16.10. Clase IPRankingList

Con los métodos de esta clase (usando los resultados de la consulta a la base de datos de la clase OpcionesDNSReverse) se genera un diagrama de barras para observar con facilidad, cuáles son las direcciones IP donde las estaciones de trabajo seleccionadas han realizado una mayor cantidad de conexiones, con el número de bytes enviados o recibidos, según sea el caso. Para conexiones TCP/UDP se muestra el número de puerto correspondiente. También hace uso de TablaDNSReverse, esta tabla muestra de mayor a menor cuáles son las direcciones IP con más actividad.

Para la resolución inversa de nombres (usando la clase ReverseIPListResolution), el usuario debe suministrar la dirección IP del servidor DNS de su red o el de su preferencia dentro del cuadro de texto indicado con la etiqueta de texto 'Ingrese la dirección IP de su servidor DNS'. En caso de que desee fijar uno por defecto, debe editar el archivo DNS_List_File.txt como se indica localizado en el directorio DNS_LIST_DIR dentro del directorio de instalación de Java y usar la palabra

default en el campo de texto destinado al servidor de nombres. La lista debe editarse como se muestra en el siguiente ejemplo: 192.188.57.242;195.5.64.6

El proceso de resolución de nombres realizará una consulta al servidor DNS en busca del nombre asociado a la dirección IP listada, en caso de no encontrarse el registro con el nombre correspondiente el valor retornado será 'No se resolvió la dirección IP'.

Incluye un botón para instanciar un objeto Browser y navegar con las direcciones IP resueltas.

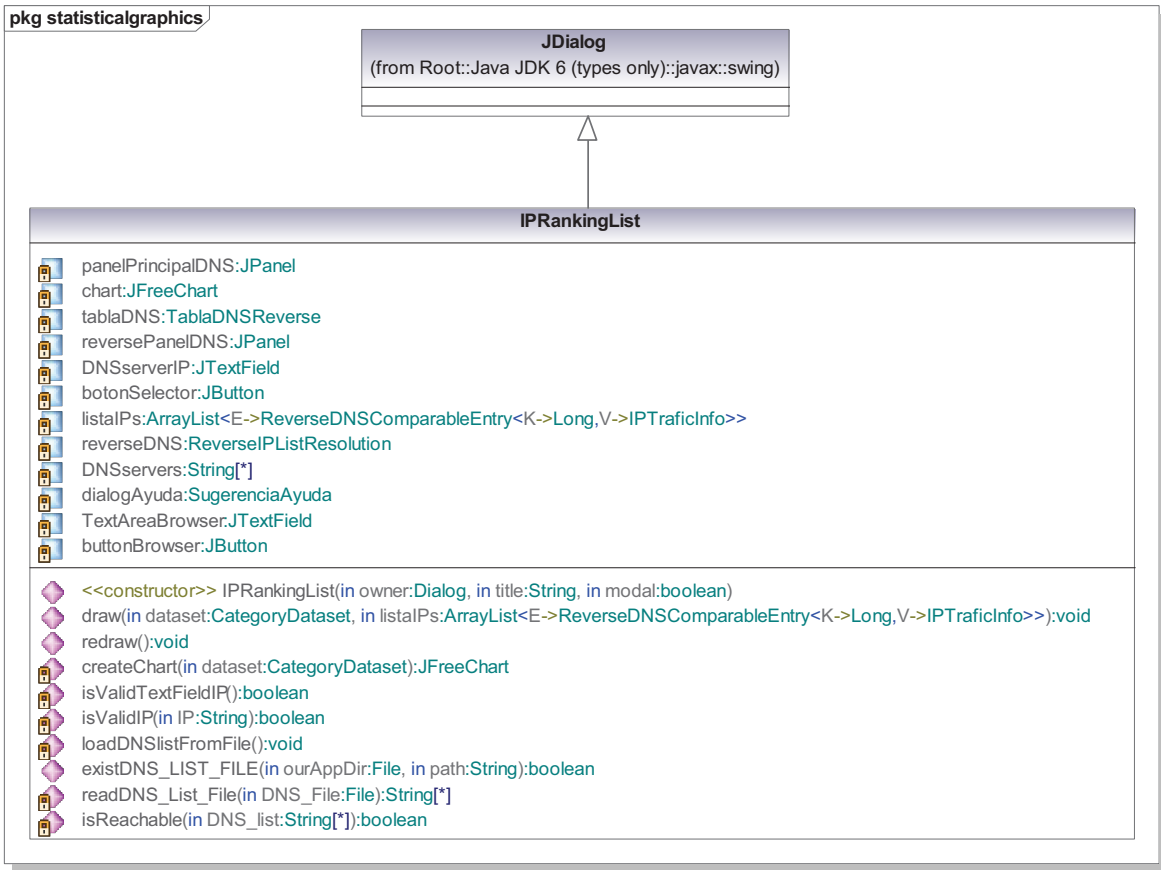


Gráfico 3.104 Clase IPRankingList

3.4.3.16.11. Clase TablaDNSReverse

La clase TablaDNSReverse permite crear una tabla gráfica con los diferentes valores de la lista de direcciones IPs para la resolución de nombres de dominio.

La tabla incluye los siguientes campos:

- Dirección IP: La representación estándar de la IP.
- IPType: Valor del campo "Protocolo" de la cabecera IP que identifica el protocolo del siguiente nivel.
- Puerto TCP/UDP: Número entre 1 - 65535 que identifica el puerto usado por la dirección IP pública para establecer la conexión.
- Bytes UP/DOWN: Cantidad en bytes de datos enviados o recibidos (depende de la opción elegida en la consulta a la base de datos).
- DNS Reverso: Nombre asociado a la dirección IP pública. Aquí se ubicarán los resultados de la resolución de nombres.

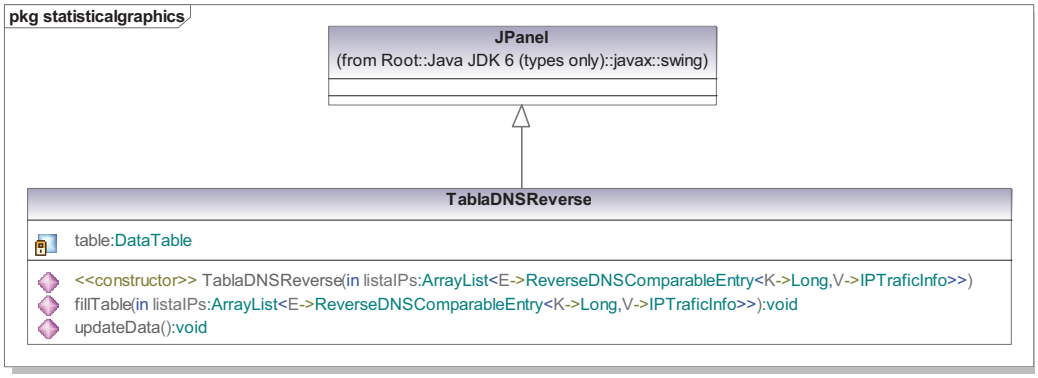


Gráfico 3.105 Clase TablaDNSReverse

3.4.3.16.12. Clase PromedioBitrateHostSeleccionados

Usando los resultados de la consulta a la base de datos (generada con las opciones de la clase OpcionesGraficoBitratePromedio), se genera un gráfico que muestra la tasa de transferencia o bitrate promedio para cada una de las estaciones de trabajo seleccionadas, por cada protocolo y/o puerto.

Para calcular estos valores se lo hace sumando la cantidad de bytes recibidos o enviados, según sea el caso, para el lapso de tiempo seleccionado en la ventana de opciones.

Esto permite establecer fácilmente quien registra la mayor tasa de transferencia y la magnitud del valor promedio. La ventana contiene una barra de desplazamiento

que permite mover la imagen en caso de que exceda el tamaño predefinido para una sola estación de trabajo.

Adjunto a esto se tiene una lista en formato de texto, ordenado de mayor a menor tasa de transferencia, con la misma información del gráfico que puede ser copiado al portapapeles seleccionando el texto requerido y usando Ctrl+C.

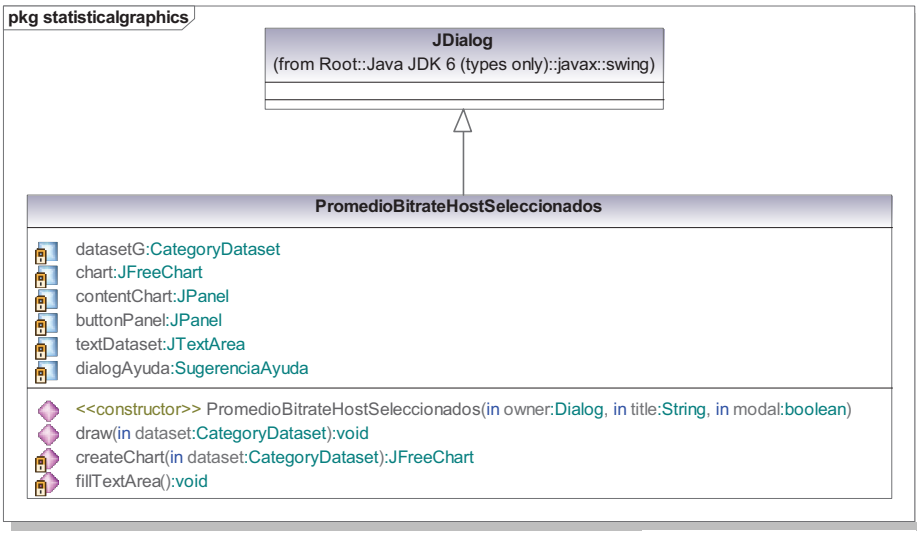


Gráfico 3.106 Clase PromedioBitrateHostSeleccionados

3.4.3.16.13. Clase PortAverageEntry

La clase PortAverageEntry permite instanciar un objeto que asocia el valor de un puerto TCP/UDP con la cantidad de bytes enviados o recibidos. Una lista de objetos de esta clase describiría el tráfico de una estación de trabajo. Al implementar un método de comparación la lista puede ser ordenada.

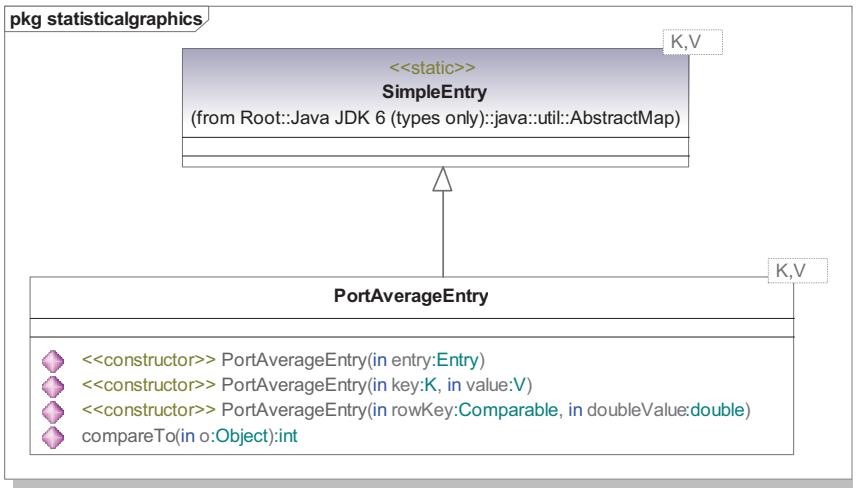


Gráfico 3.107 Clase PortAverageEntry

3.4.3.16.14. Clase BitratevsTiempoGraphLineas

Aquí se recibe el resultado de la consulta a la base de datos, construida con la clase OpcionesBitratevsTiempoProtocolo y se genera el gráfico de reconstrucción del tráfico capturado.

Una ventaja de poseer una base de datos de datos donde se almacenan periódicamente valores de tráfico de Internet, es que se puede reconstruir en cualquier momento un gráfico que muestre los valores de tasa de transferencia para las estaciones de trabajo seleccionadas diferenciando protocolos y/o puertos mediante líneas de diferente color y etiquetas de identificación. El eje horizontal representa el tiempo como la diferencia, medida en milisegundos, entre el tiempo actual y la medianoche de Enero 1 de 1970 UTC.

El usuario del programa puede observar en que momentos del día se registra un mayor uso de la conexión a Internet y si se lo hace de manera adecuada, de acuerdo a las políticas internas de la empresa o institución.

Cada línea de distinto color representa a un protocolo o tráfico asociado a un puerto, y cuando la cantidad de etiquetas de identificación sea mayor a 20 el gráfico se muestra sin ellas. En este caso, el usuario debe mover el mouse sobre los picos de las líneas y aparecerá un tooltip o texto emergente con la información necesaria para identificar cada una de ellas.

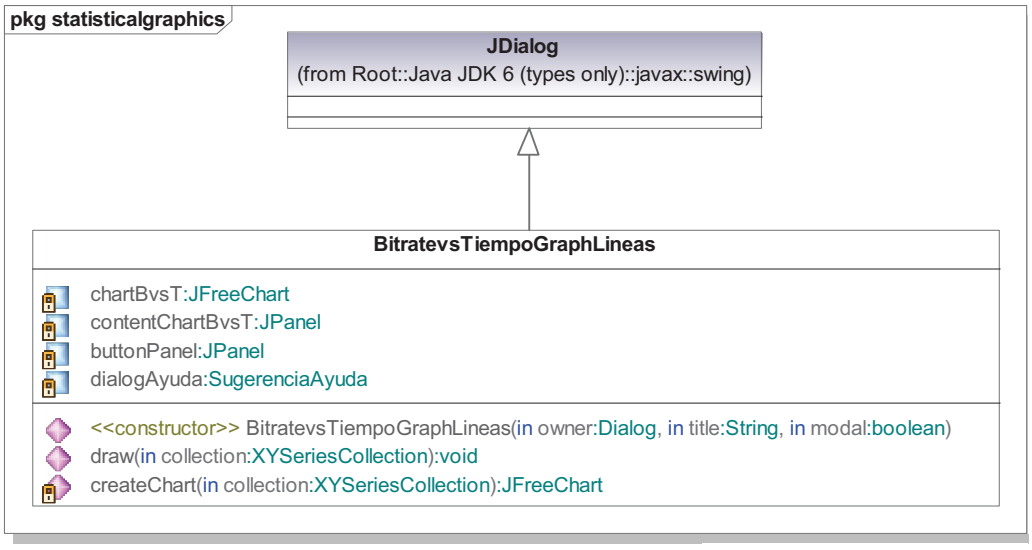


Gráfico 3.108 Clase BitratevsTiempoGraphLineas

3.4.3.16.15. Clase BitratevsTiempoGraphPasos

Implementa un gráfico similar al de BitratevsTiempoGraphLineas, la diferencia consiste en utilizar líneas horizontales en lugar de líneas, para cada valor de tasa de transferencia recuperado de la base de datos.

La altura de cada paso representa el valor promedio de tasa de transferencia y su longitud la duración de tiempo. En el eje horizontal la escala de tiempo usa el formato hh:mm (horas y minutos) para un día de 24 horas.

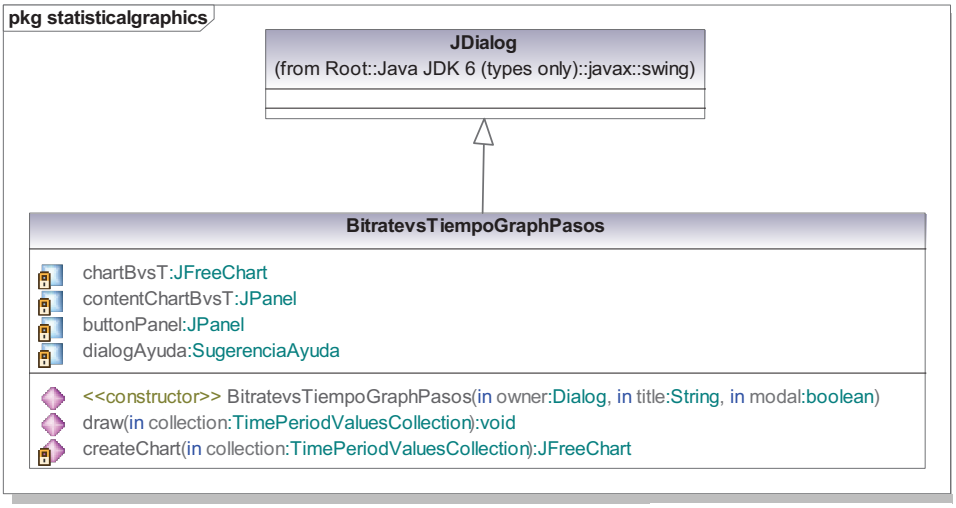


Gráfico 3.109 Clase BitratevsTiempoGraphPasos

3.4.3.16.16. Clase PercentagePieGraph2D

Esta clase recibe de OpcionesPorcentajeGraficoPastel2D los datos tras la consulta a la base de datos con las opciones seleccionadas por el usuario del programa. Usando estos datos se genera un pastel 2D para visualizar esta información.

En este gráfico pueden observarse con facilidad, de manera porcentual, los protocolos y puertos más utilizados en el acceso a Internet, de acuerdo a la cantidad de bytes enviados o recibidos, según sea el caso, para los hosts seleccionados.

El usuario del programa debe asegurarse que los protocolos y/o puertos que aparezcan en la gráfica, pertenezcan a tráfico permitido de acuerdo a las políticas de uso adecuado de la conexión a Internet de la empresa o institución.

Para facilitar la observación de las etiquetas con los porcentajes de cada sección del pastel, se ha incluido dos botones para rotar la imagen y obtener una mejor visibilidad.

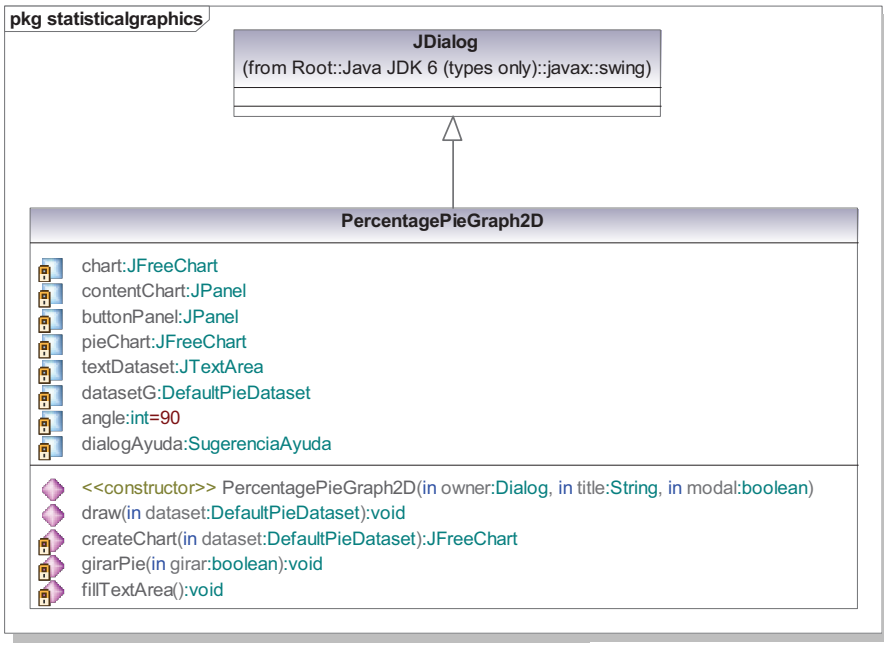


Gráfico 3.110 Clase PercentagePieGraph2D

3.4.3.16.17. Clase PercentagePieGraph3D

Es idéntica a la clase anterior con la diferencia de que se genera un pastel de porcentajes 3D, para visualizar esta información de manera más elegante y sofisticada que un gráfico 2D.

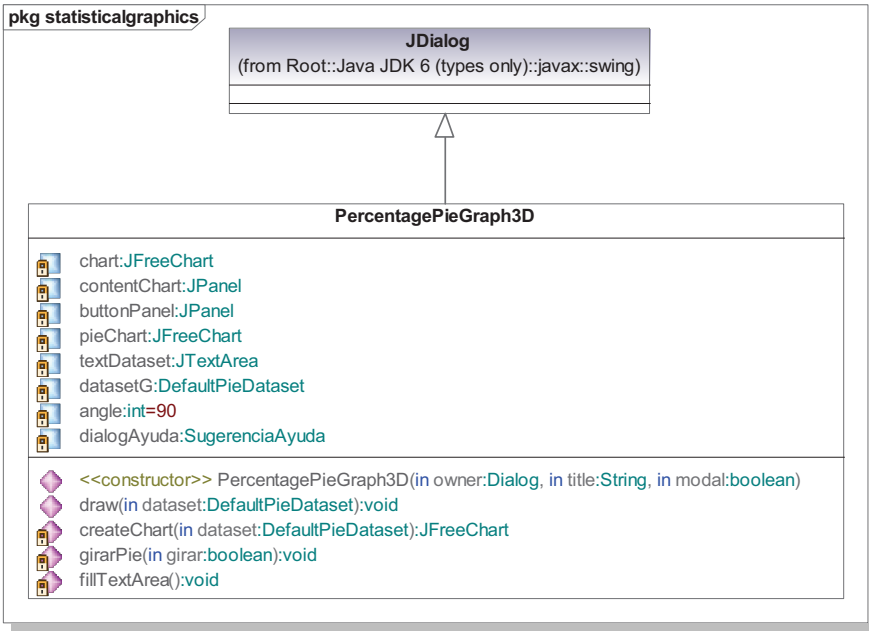


Gráfico 3.111 Clase PercentagePieGraph3D

3.4.3.16.18. Clase SerieDeTiempoGraficoLineas

Una serie de tiempo recopila de manera ordenada un conjunto de observaciones realizadas en momentos específicos y en intervalos de tiempo regulares o de igual duración, que pueden ser minutos, horas, días, meses y años. En el presente caso cada observación corresponde al valor promedio de tasa de transferencia para cada intervalo regular de tiempo.

El diálogo OpcionesSeriesTiempoProtocolos genera una consulta a la base de datos y organiza el resultado en una colección de series de tiempo. Cada serie de tiempo tiene un color específico y representa a un protocolo o puerto de una estación de trabajo, según sea el caso.

El usuario del software puede observar estos promedios gráficamente y determinar, por simple observación, dónde y en qué momento se registra la mayor cantidad de tráfico de Internet. Este tipo de gráfico tiene mayor relevancia a mediano y largo plazo, porque se dispone de mayor cantidad de datos, haciéndose más fiable contra valores de tráfico pico.

Cada serie de tiempo tiene una etiqueta identificadora. El límite de etiquetas identificadoras es de 20, si se sobrepasa este valor, no se las mostrará, con esto se evita que el espacio destinado para el gráfico disminuya demasiado. La utilidad de tooltip o texto emergente, evita que se imposibilite conocer a quien corresponde una serie de tiempo específica, si las etiquetas están ocultas.

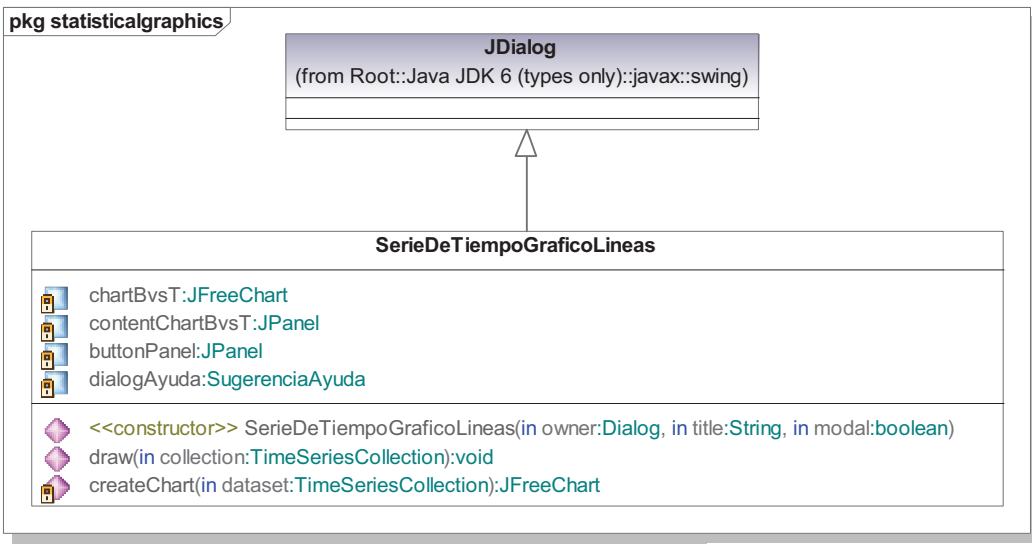


Gráfico 3.112 Clase SerieDeTiempoGraficoLineas

3.4.3.16.19. Clase SerieDeTiempoGraficoPasos

A diferencia de una serie de tiempo convencional, la graficación de pasos indica visualmente el valor promedio en cada intervalo regular, por medio de una línea horizontal. Trabaja conjuntamente con OpcionesPasosPromedioProtocolos para la consulta a la base de datos.

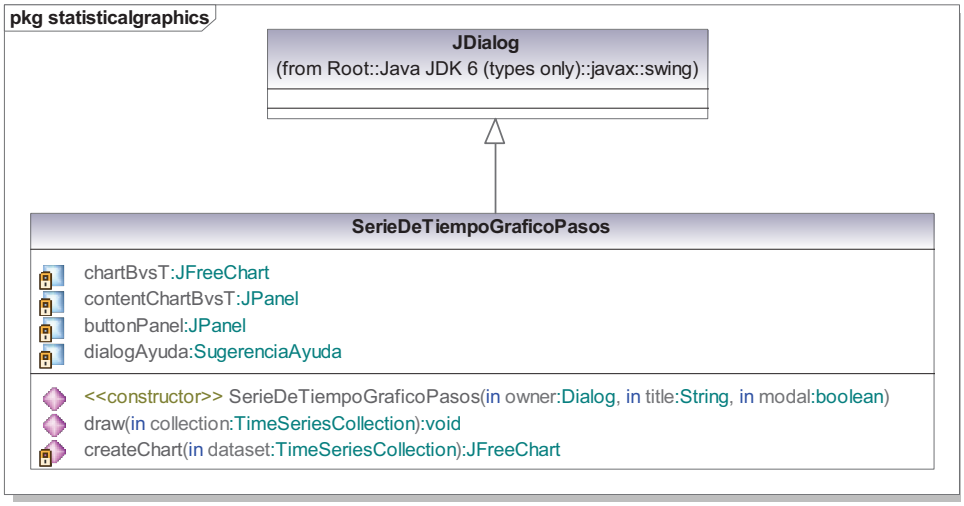


Gráfico 3.113 Clase SeriesDeTiempoGraficoPasos

Una serie de tiempo que usa líneas horizontales ayuda a evitar confusiones. Por ejemplo, el intervalo de tiempo regular es de horas; en un gráfico normal se unen con líneas los valores de los promedios de tasa de transferencia para cada intervalo, como se muestra en el gráfico 3.114.

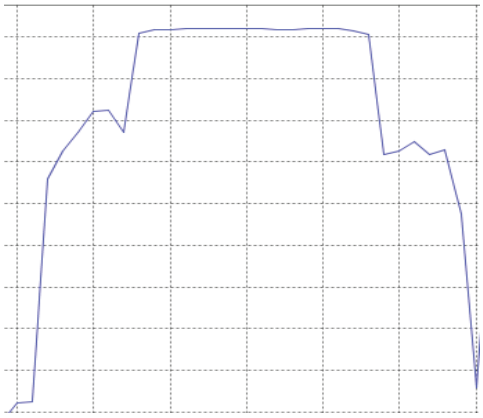


Gráfico 3.114 Fragmento de una serie de tiempo convencional.

A simple vista parecería, que al inicio de la hora, el bitrate tuvo un valor fijo y que luego aumentó o disminuyó, dependiendo del siguiente valor para el intervalo siguiente. En el gráfico 3.115 que utiliza pasos, las cosas son más claras; cada línea horizontal indica que en ese lapso de tiempo se tuvo ese valor promedio de bitrate.

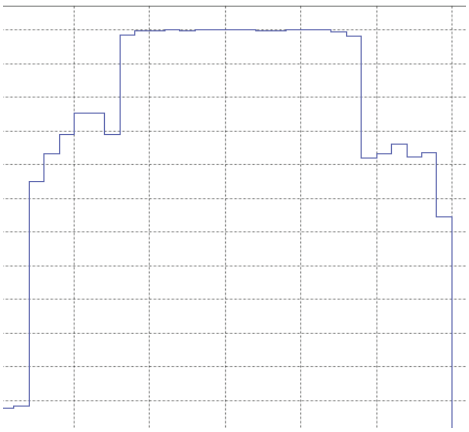


Gráfico 3.115 Fragmento de una serie de tiempo dibujada en pasos.

3.4.3.17. Clases del paquete sugerencia

3.4.3.17.1. Clase HTMLResources

La clase HTMLResources contiene un conjunto de métodos que retornan un objeto URL correspondiente a cada uno de los archivos html de ayuda y sugerencias, para los diferentes diagramas estadísticos y resúmenes de datos.

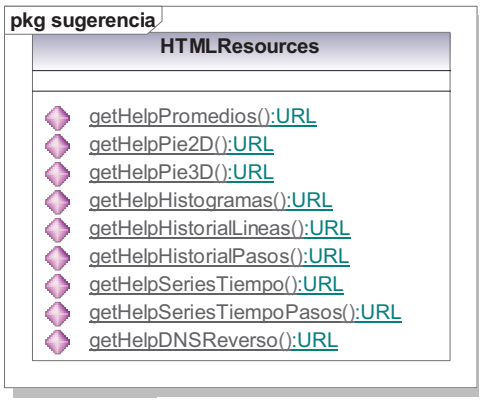


Gráfico 3.116 Clase HTMLResources

3.4.3.17.2. Clase SugerenciaAyuda

La clase SugerenciaAyuda instancia un cuadro de diálogo y un editor de contenido para mostrar la información de un archivo html guardado en el paquete de la aplicación, que proporciona una ayuda de uso e interpretación del programa para el usuario. Trabaja en conjunto con HTMLResources.

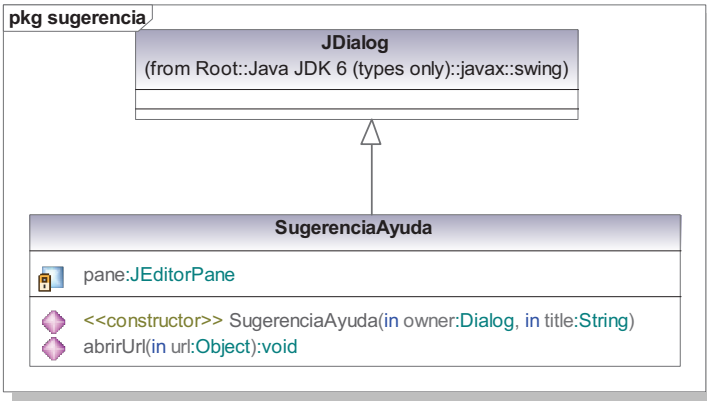


Gráfico 3.117 Clase SugerenciaAyuda

3.5. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA DE CAPTURA DE PAQUETES

El desarrollo de una interfaz gráfica permite que el usuario interactúe de manera intuitiva con el programa, sin embargo esta puede ser bastante compleja para ser escrita a partir de cero. Para disminuir esta complejidad en el presente proyecto se optó por utilizar el NetBeans IDE 6.5 con la plantilla inicial básica de una *aplicación de escritorio Java*.

3.5.1. DESCRIPCIÓN DE LOS PAQUETES DE CLASES

La aplicación contiene un paquete principal denominado *trafficstatistics* que abarca todo el conjunto de clases necesarias para la implementación de la interfaz gráfica de captura de paquetes:

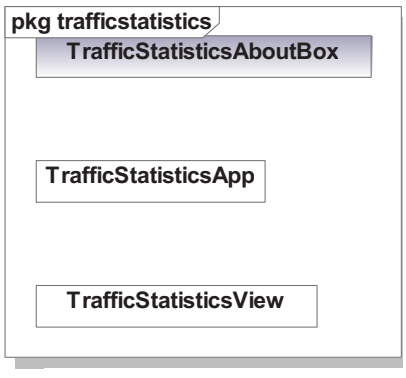


Gráfico 3.118 Paquete trafficstatistics

3.5.2. DESCRIPCIÓN DE CLASES

3.5.2.1. Clase TrafficStatisticsAboutBox

Diálogo con la información para la opción “Acerca de…”.

Gráfico 3.119 Vista previa de ventana “Acerca de…”.

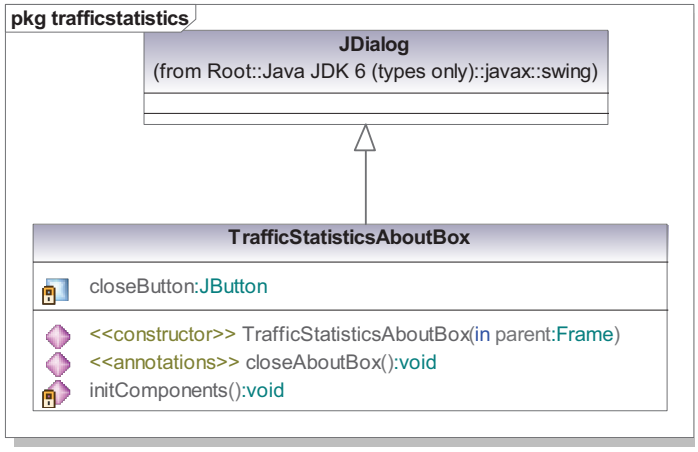


Gráfico 3.120 Clase TrafficStatisticsAboutBox

3.5.2.2. Clase TrafficStatisticsApp

Clase principal de la aplicación. Cuando se ejecuta el método principal crea una instancia de la clase de la interfaz gráfica TrafficStatisticsView.

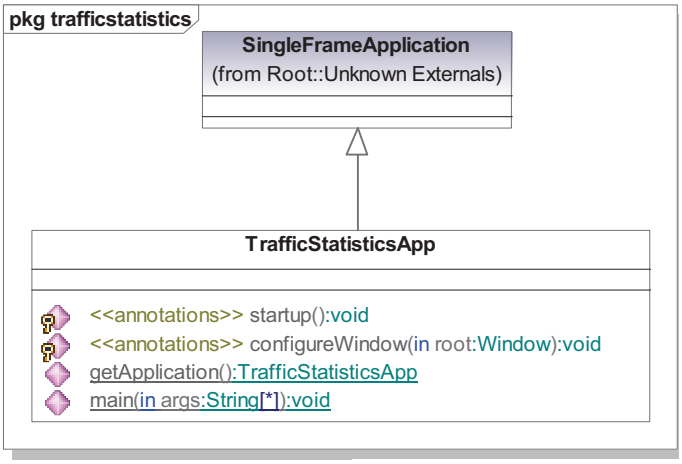


Gráfico 3.121 Clase TrafficStatisticsApp

3.5.2.3. Clase TrafficStatisticsView

La clase TrafficStatisticsView permite inicializar todos los objetos que componen toda la interfaz gráfica, tanto para la captura, graficación y almacenamiento de datos, al monitorear el tráfico de Internet de una intranet.

Dada la extensión de la clase de la interfaz gráfica TrafficStatisticsView se ha omitido su diagrama de clase, sin embargo en el siguiente punto se describe la funcionalidad que tiene cada uno de sus elementos.

3.5.3. DISEÑO E IMPLEMENTACIÓN DE LA INTERFAZ GRÁFICA UTILIZANDO EL NETBEANS IDE 6.5

A continuación se muestra la interfaz desarrollada utilizando la herramienta gráfica de diseño del IDE. Cada gráfico contiene varios círculos con un número para identificar y explicar los distintos componentes gráficos utilizados.

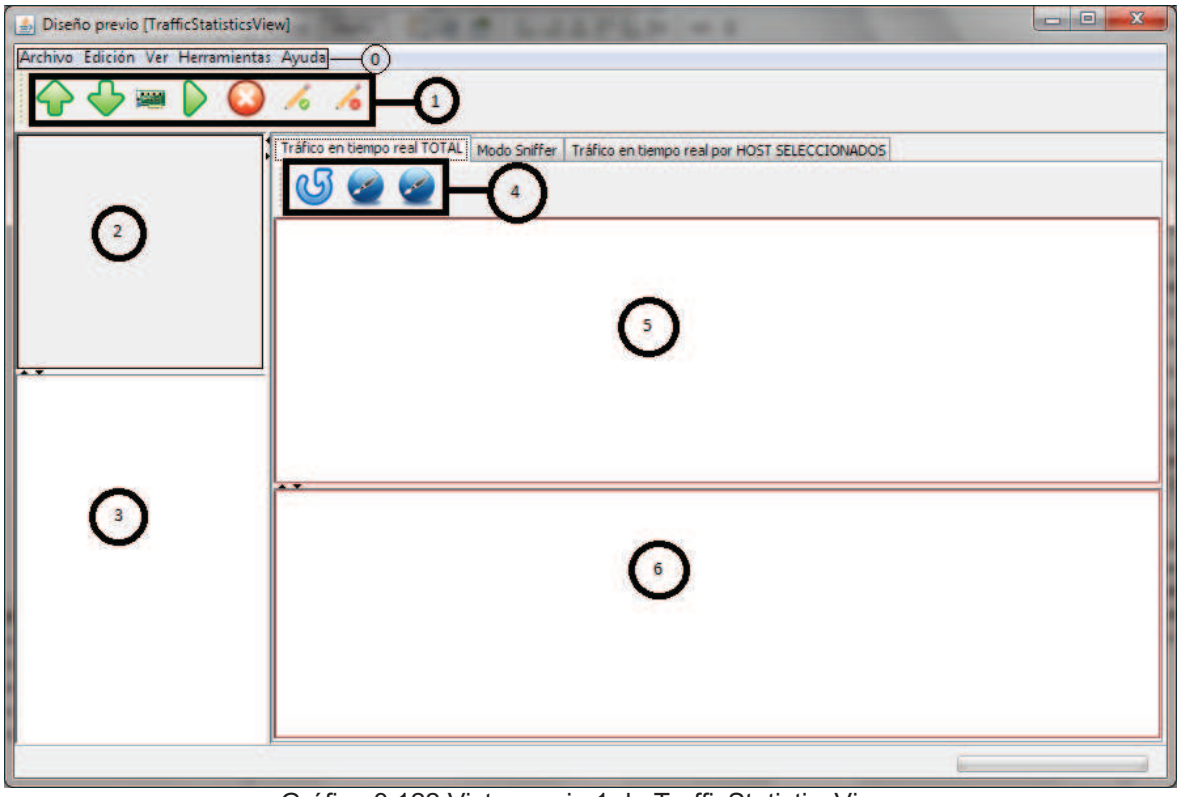


Gráfico 3.122 Vista previa 1 de TrafficStatisticsView

Identificador	Componente	Descripción
0	JMenuBar	Barra de menú con los siguientes elementos: -Archivo: Incluye + Importar Base de Datos – desde un archivo zip creado por este programa o una copia del mismo. + Exportar Base de Datos – a un archivo zip. + Salir – Finaliza la aplicación. -Edición: Incluye + Copiar Ctrl+C – copia el texto de la salida del sniffer. -Ver: Incluye + Look & Feel – despliega un cuadro de diálogo con una lista desplegable de distintos temas de apariencia para la aplicación. -Herramientas: Incluye + Información de Interfaz de Red – muestra información de la tarjeta de red seleccionada. + Selección de valor máximo de bitrate total – asigna el valor máximo de bitrate para los gráficos del tráfico de entrada y salida totales a partir del cual una se activará un sonido del sistema para alertar que se ha sobrepasado el límite permitido y los gráficos en tiempo real del tráfico modificarán su gradiente de color para este nuevo valor. + Selección de valor máximo de bitrate por hosts seleccionados – la misma funcionalidad de la opción anterior pero para los gráficos de entrada y salida de los hosts seleccionados en el sniffer. -Ayuda: Incluye + Acerca de – información sobre la aplicación.
1	JToolBar	Barra de herramientas con los siguientes botones, de izquierda a derecha respectivamente: - Exportar Base de Datos – a un archivo zip. -Importar Base de Datos – desde un archivo zip creado por este programa o una copia del mismo. -Selección de dispositivo de red – permite elegir el dispositivo deseado de una lista desplegable. -Iniciar la captura de paquetes – Abre una conexión con la base de datos donde se almacenarán los valores de bitrate, inicia la graficación del bitrate en tiempo real y la captura de paquetes para el cálculo de estos valores. -Detener la captura de paquetes – finaliza todas acciones iniciadas por el botón anterior. -Seleccionar todos los hosts detectados – selecciona todas las estaciones de trabajo detectadas durante el proceso de captura de paquetes.

		-Quitar selección de todos los hosts detectados – no selecciona ninguna estación de trabajo.
2	JScrollPane	Panel de desplazamiento donde se incluirá un árbol con las distintas estaciones de trabajo detectadas más una casilla de activación, que servirá como distintivo para saber si el host fue seleccionado o no en el momento de ejecución del sniffer.
3	JPanel	Panel que contendrá un navegador web embebido en la aplicación.
	Tráfico en tiempo real TOTAL	Primera pestaña de un componente JTabbedPane o panel con pestañas. Los componentes 4, 5 y 6 están contenidos dentro de la primera pestaña.
4	JToolBar	Barra de herramientas con los siguientes botones, de izquierda a derecha respectivamente: -Reiniciar gráficos de tráfico de Internet en tiempo real globales. -Guardar imagen de tráfico entrante total – abre un cuadro de diálogo para seleccionar el directorio y el nombre del archivo donde se guardará la imagen en el formato deseado. -Guardar imagen de tráfico saliente total – igual al anterior.
5	JPanel	Panel que contendrá el gráfico de tráfico de Internet entrante en tiempo real global, es decir, de todo las estaciones de trabajo activas.
6	JPanel	Igual al anterior pero para tráfico saliente.

Tabla 3.3 Descripción de la vista previa 1 de TrafficStatisticsView

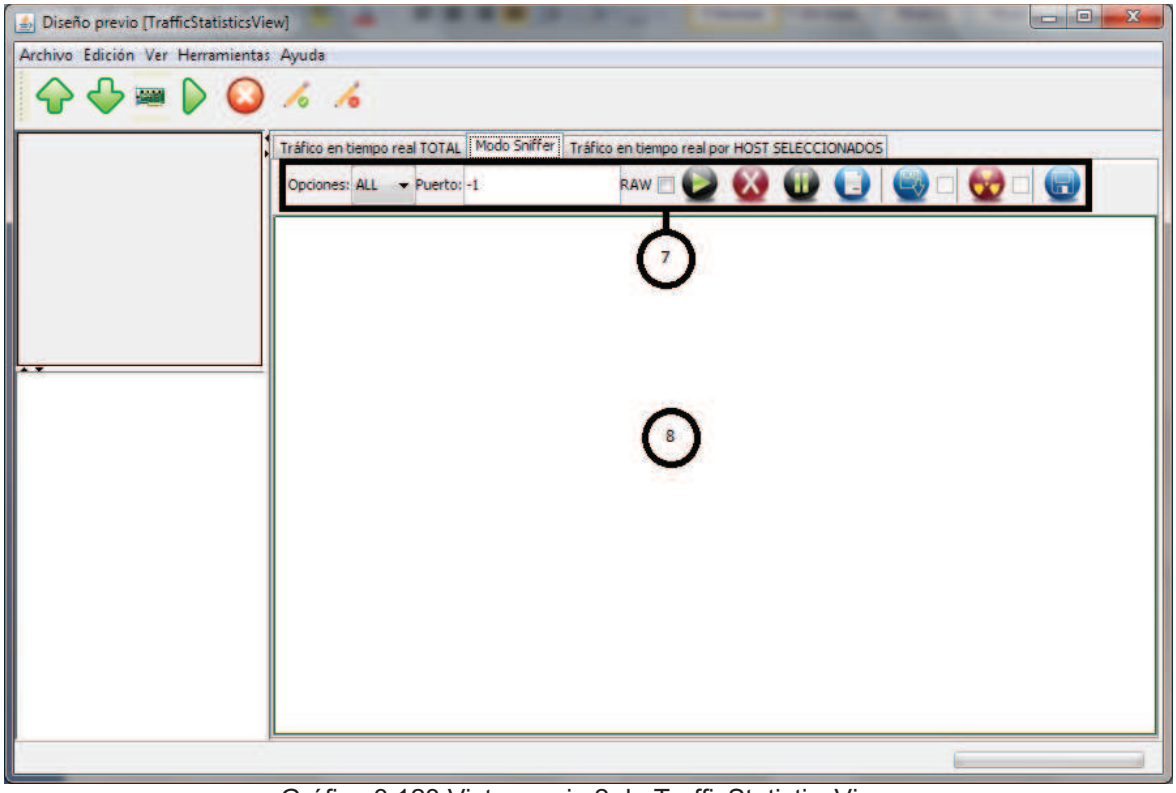


Gráfico 3.123 Vista previa 2 de TrafficStatisticsView

Identificador	Componente	Descripción
	Modo Sniffer	Segunda pestaña de un componente JTabbedPane o panel con pestañas. Los componentes 7 y 8 están contenidos dentro de la primera pestaña.
7	JToolBar	Barra de herramientas con los siguientes componentes, de izquierda a derecha respectivamente: -Opciones de protocolos – lista desplegable detallada a continuación: ALL-->Todos los Paquetes TCP-->Paquetes que usan el Transmission Control Protocol como protocolo de transporte UDP-->Paquetes que usan el User Datagram Protocol como protocolo de transporte ICMP-->Paquetes provenientes del Internet Control Message Protocol -Puerto – campo de texto para el ingreso de un valor entre 1 y 65535 que representa los valores de puertos para los protocolos de transporte TCP y UDP. En el caso de ICMP el contenido de este campo es ignorado. -RAW – casilla de activación, si está seleccionada el paquete será decodificado completamente, caso contrario se mostrarán decodificadas solo las cabeceras de los protocolos. -Iniciar o continuar la ejecución del sniffer – inicia la funcionalidad del sniffer con los parámetros seleccionados, conjuntamente inician los gráficos para el tráfico de los hosts seleccionados que utilizan los mismos parámetros. -Detener el sniffer – detiene la ejecución del sniffer y finaliza la graficación para el tráfico de los hosts seleccionados. -Pausar el sniffer – pausa la ejecución del sniffer. -Borrar contenido – borra el contenido de los paquetes decodificados del área de texto. -Guardar captura automáticamente – abre un cuadro de diálogo para seleccionar el directorio y el nombre del archivo donde se guardará la captura de paquetes decodificados en formato de texto. -No guardar captura automáticamente – el contenido del sniffer no se guardará automáticamente y se reiniciará cada 15000 líneas. -Guardar texto actual – abre una ventana de diálogo para guardar el contenido del área de texto del sniffer.
8	JTextArea	Área de texto donde se despliega el contenido decodificado de los paquetes capturados.

Tabla 3.4 Descripción de la vista previa 2 de TrafficStatisticsView

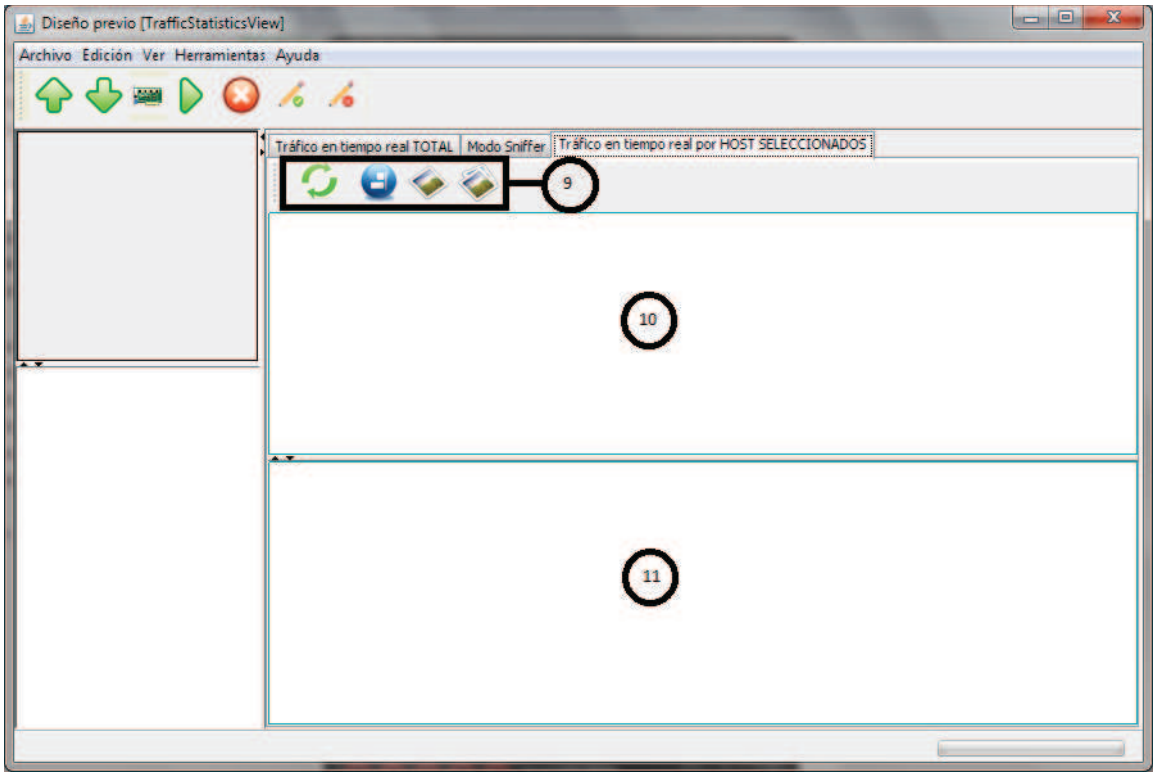


Gráfico 3.124 Vista previa 3 de TrafficStatisticsView

Identificador	Componente	Descripción
	Tráfico en tiempo real por HOSTS SELECCIONADOS	Segunda pestaña de un componente JTabbedPane o panel con pestañas. Los componentes 9,10 y 11 están contenidos dentro de la primera pestaña.
9	JToolBar	Barra de herramientas con los siguientes botones, de izquierda a derecha respectivamente: -Reiniciar gráficos de tráfico de Internet en tiempo real de hosts seleccionados. -Sincronizar gráficos globales y de hosts seleccionados – ambos tipos de gráficos inician nuevamente al mismo tiempo. -Guardar imagen de tráfico entrante de los hosts seleccionados - abre un cuadro de diálogo para seleccionar el directorio y el nombre del archivo donde se guardará la imagen en el formato deseado. -Guardar imagen de tráfico saliente total de los hosts seleccionados – igual al anterior.
10	JPanel	Panel que contendrá el gráfico de tráfico de Internet entrante en tiempo real de los hosts seleccionados, es decir, de todas las estaciones de trabajo activas seleccionadas.
11	JPanel	Igual al anterior pero para tráfico saliente.

Tabla 3.5 Descripción de la vista previa 3 de TrafficStatisticsView

3.6. DESCRIPCIÓN DE LA INTERFAZ GRÁFICA DE ANÁLISIS ESTADÍSTICO

Al igual que en la captura de datos, en la interfaz gráfica para la aplicación de escritorio de análisis estadístico se utilizó el NetBeans IDE 6.5.

3.6.1. DESCRIPCIÓN DE LOS PAQUETES DE CLASES

La aplicación contiene un paquete principal denominado *querystatistics* con el conjunto de clases necesarias para la implementación de la interfaz gráfica de análisis estadístico:

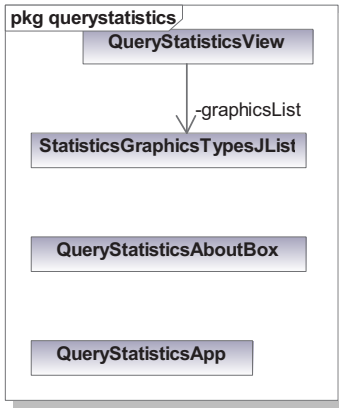


Gráfico 3.125 Paquete querystatistics

3.6.2. DESCRIPCIÓN DE CLASES

3.6.2.1. Clase QueryStatisticsAboutBox

Diálogo con la información para la opción “Acerca de…”.

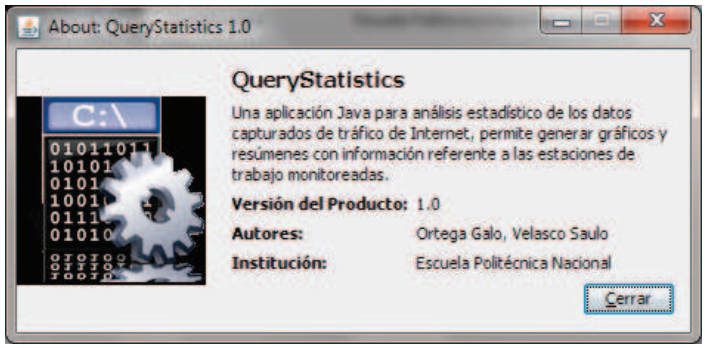


Gráfico 3.126 Vista previa de ventana “Acerca de…”.

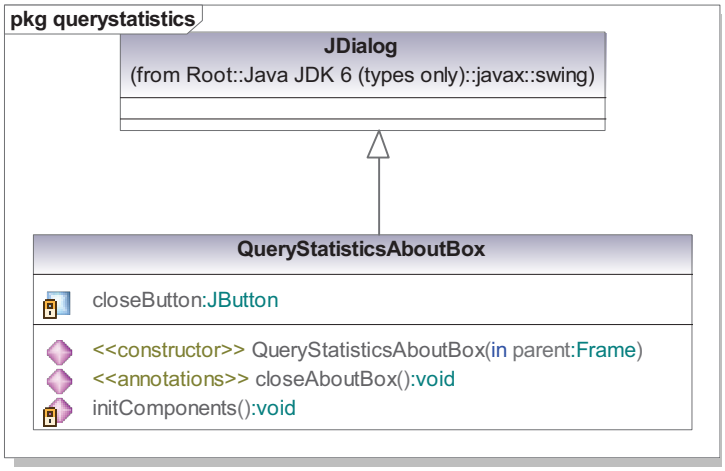


Gráfico 3.127 Clase QueryStatisticsAboutBox

3.6.2.2. Clase QueryStatisticsApp

Clase principal de la aplicación. Cuando se ejecuta el método principal crea una instancia de la clase de la interfaz gráfica QueryStatisticsView.

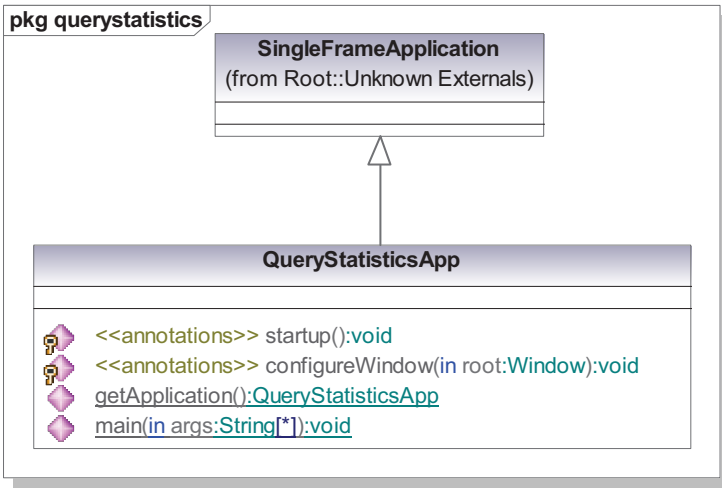


Gráfico 3.128 Clase QueryStatisticsApp

3.6.2.3. Clase StatisticsGraphicsTypesJList

La clase StatisticsGraphicsTypesJList es una lista visual que permite mostrar los diferentes tipos de gráficos y resúmenes de datos que el programa puede generar. Para acceder a una ventana de diálogo de cada uno de los diagramas, incluye la implementación de escucha para el evento doble clic, además permite cargar la información y animación respectiva de la opción elegida por el usuario.

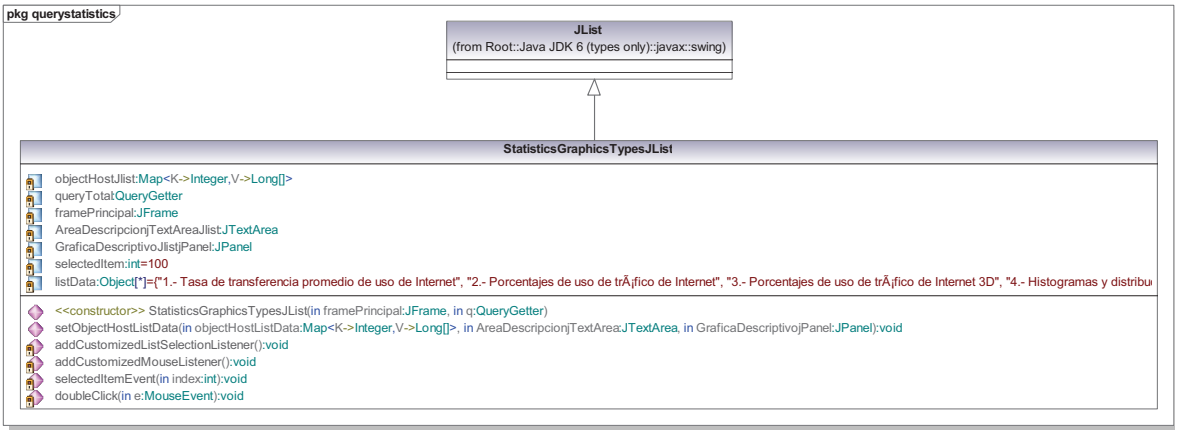


Gráfico 3.129 Clase StatisticsGraphicsTypesJList

3.6.2.4. Clase QueryStatisticsView

La clase QueryStatisticsView permite inicializar todos los objetos que componen toda la interfaz gráfica, tanto para la representación de los distintos diagramas estadísticos y resúmenes, para el establecimiento y finalización de la conexión a la base de datos y la descripción de cada uno de ellos con su respectiva animación informativa.

Al igual que la interfaz gráfica para captura de datos, la de análisis estadístico QueryStatisticsView también es muy extensa. En consecuencia su diagrama de clase ha sido omitido e igualmente se procede a describir cada uno de sus elementos gráficos.

3.6.3. DISEÑO E IMPLEMENTACIÓN DE LA INTERFAZ GRÁFICA UTILIZANDO EL NETBEANS IDE 6.5

Siguiendo la modalidad para la interfaz gráfica anterior, se mostrarán las capturas de los diseños realizados con el IDE y su explicación en forma tabulada.

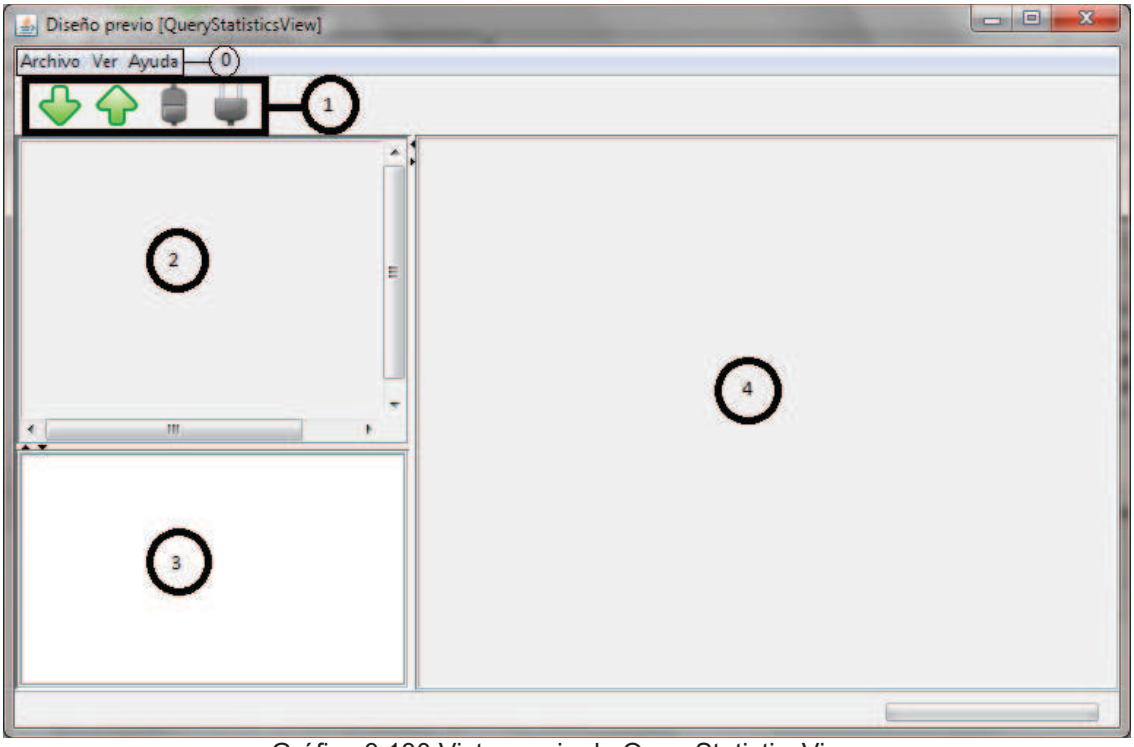


Gráfico 3.130 Vista previa de QueryStatisticsView

Identificador	Componente	Descripción
0	JMenuBar	Barra de menú con los siguientes elementos: -Archivo: Incluye + Importar Base de Datos – desde un archivo zip creado por este programa o una copia del mismo. + Exportar Base de Datos – a un archivo zip. + Salir – finaliza la aplicación. -Ver: Incluye + Look & Feel – despliega un cuadro de diálogo con una lista desplegable de distintos temas de apariencia para la aplicación. -Ayuda: Incluye + Acerca de – información sobre la aplicación.
1	JToolBar	Barra de herramientas con los siguientes botones, de izquierda a derecha respectivamente: -Importar Base de Datos – desde un archivo zip creado por este programa o una copia del mismo. -Exportar Base de Datos – a un archivo zip. -Conectar a Base de Datos – establece una conexión con la base de datos para la realización de consultas. -Desconectar de Base de Datos – finaliza la conexión de la base de datos de manera segura.
2	JPanel	Panel contenido en un panel de desplazamiento donde se ubicará un StatisticsGraphicsTypesJList, que es una clase personalizada que hereda

		todos los atributos y funcionalidad de un JList o lista. Esta lista contiene todos los gráficos y resúmenes de los datos capaces de obtener por la aplicación y se inicializa inmediatamente después de establecer la conexión con la base de datos. Para acceder al cuadro de diálogo para la generación de los gráficos se hace doble clic sobre el nombre del gráfico deseado.
3	JTextArea	Área de texto donde se incluye una descripción textual de los gráficos. Cambia de acuerdo a la selección hecha en la lista.
4	JPanel	Panel donde se reproduce una animación flash por cada tipo gráfico seleccionado.

Tabla 3.6 Descripción vista previa de QueryStatisticsView

3.7. DESCRIPCIÓN DE CASOS DE USO

A continuación se describirá el desarrollo de los casos de uso del gráfico 3.1, especificando sus detalles y condiciones para las aplicaciones TrafficStatistics y QueryStatistics.

3.7.1. CASOS DE USO REALIZADOS POR LA APLICACIÓN TRAFFIC STATISTICS

Caso de Uso: Graficar tráfico total en tiempo real	
Actor: Administrador	
Descripción: Permite mostrar gráficamente el tráfico total entrante y saliente en tiempo real para el dispositivo de red elegido por el administrador.	
Activación: Realizar un clic en el botón “iniciar la captura de paquetes”	
Curso Normal	**Alternativas**
1.- Inicia la aplicación TrafficStatistics	Ninguna
2.- Elegir el dispositivo de red a ser monitoreado.	Ninguna
3.-Iniciar el monitoreo y representación gráfica del tráfico de Internet con el botón “iniciar la captura de paquetes”	Ninguna
Precondiciones: Haber elegido la tarjeta de red en donde se va a monitorear el tráfico entrante como saliente.	
Postcondiciones: Ninguna	
Puntos de extensión: Ninguna	
Observaciones y Datos: Ninguna	

Tabla 3.7 Caso de uso Graficar tráfico total en tiempo real para el actor Administrador.

Caso de Uso: Desplegar un sniffer y guardar contenido	
Actor: Administrador	
Descripción: Permite mostrar un sniffer y guardar su contenido para el tráfico monitoreado en tiempo real por hosts seleccionados para el dispositivo de red elegido por el administrador.	
Activación: Realizar un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	
Curso Normal	**Alternativas**

1.- Inicia la aplicación TrafficStatistics	Ninguna
2.- Elegir el dispositivo de red a ser monitoreado.	Ninguna
3.-Iniciar el monitoreo y representación gráfica del tráfico de Internet con el botón “iniciar la captura de paquetes”	Ninguna
4.- Elegir la pestaña “Modo sniffer”en la parte derecha de la pantalla.	Ninguna
5.- Seleccionar los hosts a monitorear del árbol de estaciones de trabajo.	Ninguna
6.-Seleccionar un protocolo sea del tipo TCP, UDP, ICMP o Todos	6.1.-No seleccionar ningún protocolo ya que esta por defecto asignado todos los protocolos.
7.-Seleccionar un puerto para el filtro, teniendo en cuenta que todos los puertos está indicado con el valor -1.	6.1.-No seleccionar ningún puerto ya que esta por defecto asignado todos los puertos con el valor -1.
7.- Inicializar el análisis realizando un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	Ninguna
Precondiciones: Haber elegido la tarjeta de red en donde se va a monitorear el tráfico entrante como saliente, haber inicializado el monitoreo del tráfico total y haber seleccionado los hosts a monitorear de un árbol de estaciones de trabajo. (Opcionales): Haber elegido un protocolo TCP,UDP,ICMP o todos, además haber seleccionado un puerto y haber asignado la forma de respaldar los datos obtenidos durante el monitoreo.	
Postcondiciones: (Opcional) Seleccionar la manera y el directorio en donde se almacenarán los datos monitoreados.	
Puntos de extensión: Se puede imprimir los campos de los paquetes monitoreados en forma reducida o en forma extendida seleccionando el checkbox RAW en la barra de herramientas del sniffer.	
Observaciones y Datos: -Si no se ha seleccionado ninguna forma de almacenar datos, se desplegara un cuadro de diálogo que permitirá guardar los datos cada 15000 líneas impresas en el área de texto. -Si no se ha seleccionado ningún protocolo ni puerto está asignado por defecto todos los protocolos y todos los puertos.	

Tabla 3.8 Caso de uso Desplegar un sniffer y guardar contenido para el actor Administrador.

Caso de Uso: Graficar tráfico en tiempo real por hosts.	
Actor: Administrador	
Descripción: Permite mostrar gráficamente el tráfico entrante y saliente por hosts seleccionados en tiempo real para el dispositivo de red elegido por el administrador.	
Activación: Realizar un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado” y seleccionar la pestaña “tráfico en tiempo real por hosts seleccionados”	
Curso Normal	**Alternativas**
1.- Inicia la aplicación TrafficStatistics	Ninguna
2.- Elegir el dispositivo de red a ser monitoreado.	Ninguna
3.-Iniciar el monitoreo y representación gráfica del tráfico de Internet con el botón “iniciar la captura de paquetes”	Ninguna

4.- Elegir la pestaña “Modo sniffer”en la parte derecha de la pantalla.	Ninguna
5.- Seleccionar los hosts a monitorear del árbol de estaciones de trabajo.	Ninguna
5.-Seleccionar un protocolo sea del tipo TCP, UDP, ICMP o Todos	5.1.-NO seleccionar ningún protocolo ya que esta por defecto asignado todos los protocolos.
6.-Seleccionar un puerto para el filtro, teniendo en cuenta que todos los puertos está indicado con el valor -1.	6.1.-NO seleccionar ningún puerto ya que esta por defecto asignado todos los puertos con el valor -1.
7.- Inicializar el análisis realizando un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	Ninguna
8.-Seleccionar la pestaña “tráfico en tiempo real por hosts seleccionados”	Ninguna
Precondiciones: Haber elegido la tarjeta de red en donde se va a monitorear el tráfico entrante como saliente, haber inicializado el monitoreo del tráfico total y haber seleccionado los hosts a monitorear de un árbol de estaciones de trabajo. (Opcionales): Haber elegido un protocolo TCP,UDP,ICMP o todos, además haber seleccionado un puerto y haber asignado la forma de respaldar los datos obtenidos durante el monitoreo. Inicializar la captura realizando un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	
Postcondiciones: (Opcional) Seleccionar la manera y el directorio en donde se almacenarán los datos monitoreados.	
Puntos de extensión: Ninguna	
Observaciones y Datos: Ninguna	

Tabla 3.9 Caso de uso Graficar tráfico en tiempo real por hosts para el actor Administrador.

Caso de Uso: Diferenciar tráfico (Host, protocolo y puertos)	
Actor: Desplegar un sniffer y guardar contenido	
Descripción: Permite diferenciar el tráfico de Internet monitoreado a través de un filtro por host, protocolo y puertos.	
Activación: Realizar un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	
Curso Normal	**Alternativas**
1.- Inicia la aplicación TrafficStatistics	Ninguna
2.- Elegir el dispositivo de red a ser monitoreado.	Ninguna
3.-Iniciar el monitoreo y representación gráfica del tráfico de Internet con el botón “iniciar la captura de paquetes”	Ninguna
4.- Elegir la pestaña “Modo sniffer”en la parte derecha de la pantalla.	Ninguna
5.- Seleccionar los hosts a monitorear del árbol de estaciones de trabajo.	Ninguna
5.-Seleccionar un protocolo sea del tipo TCP, UDP, ICMP o Todos	5.1.-No seleccionar ningún protocolo ya que esta por defecto asignado todos los protocolos.
6.-Seleccionar un puerto para el filtro, teniendo en cuenta que todos los puertos está indicado con el valor -1.	6.1.-No seleccionar ningún puerto ya que esta por defecto asignado todos los puertos con el valor -1.

7.- Inicializar el análisis realizando un clic en el botón “Inicializa o continua el despliegue de los datos de acuerdo al filtro seleccionado”	Ninguna
Precondiciones: Haber elegido la tarjeta de red en donde se va a monitorear el tráfico entrante como saliente, haber inicializado el monitoreo del tráfico total y haber seleccionado los hosts a monitorear de un árbol de estaciones de trabajo. (Opcionales): Haber elegido un protocolo TCP,UDP,ICMP o todos, además haber seleccionado un puerto y haber asignado la forma de respaldar los datos obtenidos durante el monitoreo.	
Postcondiciones: Ninguna.	
Puntos de extensión: Ninguna.	
Observaciones y Datos: -Si no se ha seleccionado ningún protocolo ni puerto está asignado por defecto todos los protocolos y todos los puertos.	

Tabla 3.10 Caso de uso Diferenciar tráfico (Host, protocolo y puertos) para el actor Desplegar un sniffer y guardar contenido.

Caso de Uso: Almacenar y recuperar de disco.	
Actor: Capturar tráfico de Internet	
Descripción: Permite guardar los valores de longitud de paquete y los timestamp monitoreados en una base de datos MySQL.	
Activación: Realizar un clic en el botón “iniciar la captura de paquetes”	
Curso Normal	Alternativas
1.- Inicia la aplicación TrafficStatistics	Ninguna
2.- Elegir el dispositivo de red a ser monitoreado.	Ninguna
3.-Iniciar el monitoreo y representación gráfica del tráfico de Internet con el botón “iniciar la captura de paquetes”	Ninguna
Precondiciones: -Haber elegido la tarjeta de red en donde se va a monitorear el tráfico entrante como saliente. -Haber inicializado la captura de datos con el botón “iniciar la captura de paquetes” -Haber flujo de datos para almacenarlos.	
Pos condiciones: Ninguna	
Puntos de extensión: Ninguna	
Observaciones y Datos: Ninguna	

Tabla 3.11 Caso de uso Almacenar y recuperar de disco para el actor Capturar tráfico de Internet

3.7.2. CASOS DE USO REALIZADOS POR LA APLICACIÓN QUERY STATISTICS

Caso de Uso: Mostrar y guardar gráfico de análisis estadístico.	
Actor: Administrador	
Descripción: Permite mostrar gráficamente los resultados estadísticos obtenidos de los valores almacenados en la base de datos.	
Activación: Inicializar la aplicación Query Statistics para el análisis de los datos.	
Curso Normal	**Alternativas**
1.- Inicia la aplicación QueryStatistics	Ninguna
2.- Conectar la base de datos MySQL.	Ninguna
3.-Elegir un análisis estadístico en particular de la lista.	Ninguna
4.- Asignar los diferentes filtros de puertos, protocolos, fechas para los datos que el	

usuario desee analizar.
Precondiciones: -Haber guardado datos durante el monitoreo del flujo de tráfico a Internet. -Haber elegido la base de datos de donde se va a extraer los valores almacenados del monitoreo. -Haber inicializado la conexión a la base de datos.
Pos condiciones: (Automático) cierra la base de datos si no está usando la aplicación traffic statistics.
Observaciones y Datos: Ninguna

Tabla 3.12 Caso de uso Mostrar y guardar gráfico de análisis estadístico para el actor Administrador.

Caso de Uso: Diferenciar tráfico (Host, protocolo y puertos)	
Actor: Mostrar y guardar gráfico de análisis estadístico.	
Descripción: Permite diferenciar los valores obtenidos del tráfico de Internet almacenados en la base de datos MySQL a través de un filtro por host, protocolo y puertos.	
Activación: Inicializar la aplicación Query Statistics para el análisis de los datos.	
Curso Normal	**Alternativas**
1.- Inicia la aplicación QueryStatistics	Ninguna
2.- Conectar la base de datos MySQL.	Ninguna
3.-Elegir un análisis estadístico en particular de la lista.	Ninguna
4.- Asignar los diferentes filtros de puertos, protocolos, fechas para los datos que el usuario desee analizar.	Ninguna
Precondiciones: -Haber guardado datos durante el monitoreo del flujo de tráfico a Internet. -Haber elegido la base de datos de donde se va a extraer los valores almacenados del monitoreo. -Haber inicializado la conexión a la base de datos.	
Pos condiciones: (Automático) cierra la base de datos si no está usando la aplicación traffic statistics.	
Puntos de extensión: Ninguna.	
Observaciones y Datos: Ninguna.	

Tabla 3.13 Caso de uso Diferenciar tráfico (Host, protocolo y puertos) para el actor Mostrar y guardar gráfico de análisis estadístico.

Caso de Uso: Almacenar y recuperar de disco.	
Actor: Mostrar y guardar gráfico de análisis estadístico.	
Descripción: Permite recuperar los valores de longitud de paquete y los timestamp monitoreados de una base de datos MySQL.	
Activación: Inicializar la aplicación Query Statistics para el análisis de los datos.	
Curso Normal	**Alternativas**
1.- Inicia la aplicación QueryStatistics	Ninguna
2.- Conectar la base de datos MySQL.	Ninguna
3.-Elegir un análisis estadístico en particular de la lista.	Ninguna
4.- Asignar los diferentes filtros de puertos, protocolos, fechas para los datos que el usuario desee analizar.	Ninguna
Precondiciones:	

-Haber guardado datos durante el monitoreo del flujo de tráfico a Internet. -Haber elegido la base de datos de donde se va a extraer los valores almacenados del monitoreo. -Haber inicializado la conexión a la base de datos.
Pos condiciones: (Automático) cierra la base de datos si no está usando la aplicación traffic statistics.
Puntos de extensión: Ninguna
Observaciones y Datos: Ninguna

Tabla 3.14 Caso de uso Almacenar y recuperar de disco para el actor Mostrar y guardar gráfico de análisis estadístico.

3.8. DIAGRAMAS DE SECUENCIA Y COLABORACIÓN

3.8.1. TRAFFIC STATISTICS

Casos de uso:

- **Graficar tráfico total en tiempo real <<include>> Capturar tráfico de Internet.**

El usuario genera un evento sobre el botón de inicio de captura, que desencadena una secuencia de llamadas como se muestra en el gráfico 3.131.

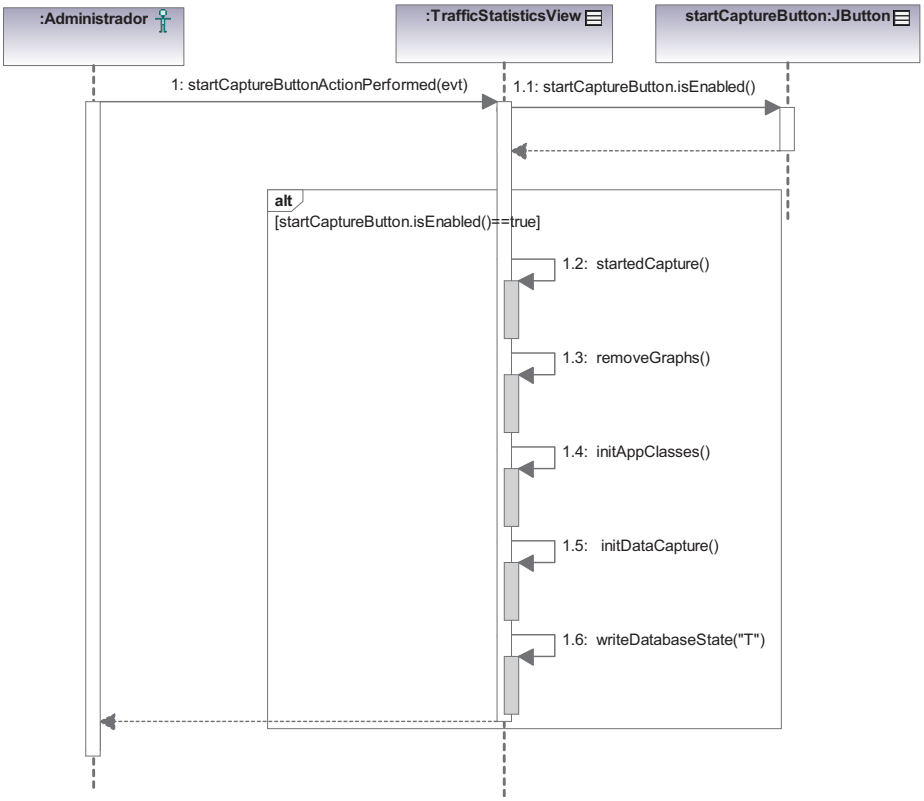


Gráfico 3.131 Diagrama de secuencia para el inicio de la graficación en tiempo real del tráfico total y captura de paquetes del método startCaptureButtonActionPerformed(…) de la clase TrafficStatisticsView.

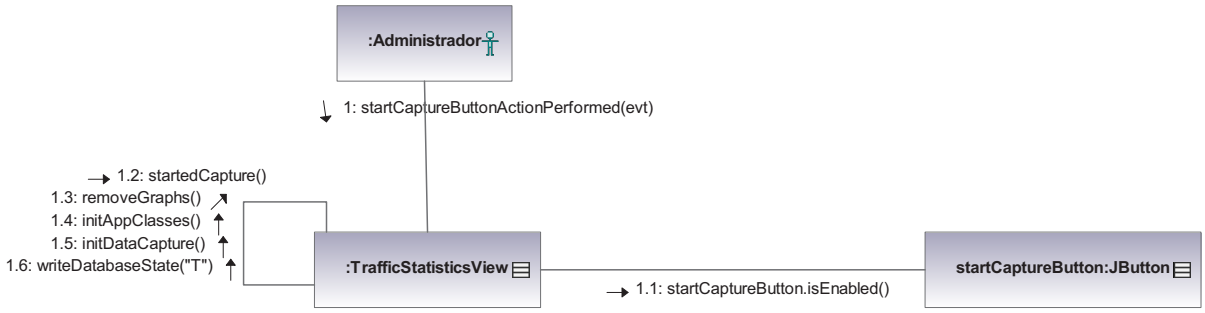


Gráfico 3.132 Diagrama de colaboración para el inicio de la graficación en tiempo real del tráfico total y captura de paquetes del método startCaptureButtonActionPerformed(…) de la clase TrafficStatisticsView.

startCaptureButtonActionPerformed(…)	
Mensaje	Significado
1: startCaptureButtonActionPerformed(evt)	Inicializa la captura, almacenamiento y graficación del tráfico de Internet, con un evento generado por el usuario.
1.1: startCaptureButton.isEnabled()	Verifica si el botón de inicio está habilitado para su uso.
1.2: startedCapture()	Estado de los botones cuando se hace clic sobre el botón de inicio de captura del tráfico de Internet.
1.3: removeGraphs()	Remueve las formas dibujadas del tráfico de Internet de los hosts seleccionados así como del tráfico total y cancela la graficación automática para todos los gráficos.
1.4: initAppClasses()	Instancia e inicializa todos los objetos necesarios para la captura, almacenamiento y graficación para el monitoreo del tráfico de Internet, así como un listener para la parte del área de texto y un timer para la actualización de la base de datos
1.5: initDataCapture()	Inicia la captura de datos con los objetos necesarios para realizar el monitoreo del tráfico de Internet.
1.6: writeDatabaseState("T")	Escribe el estado de utilización de la base de datos a un archivo para que otra aplicación pueda saber si cierra o no la conexión. Si el valor almacenado es "T" indica que TrafficStatistics tiene activa la conexión a la base de datos, "Stop" indica que la base no está siendo utilizada. Además verifica el estado de utilización de la base de datos del otro programa que usa la misma conexión.

Tabla 3.15 Mensajes del diagrama de colaboración del gráfico 3.132

Para describir con mayor profundidad la secuencia de mensajes se desarrolló el diagrama del método initDataCapture() en el gráfico 3.133.

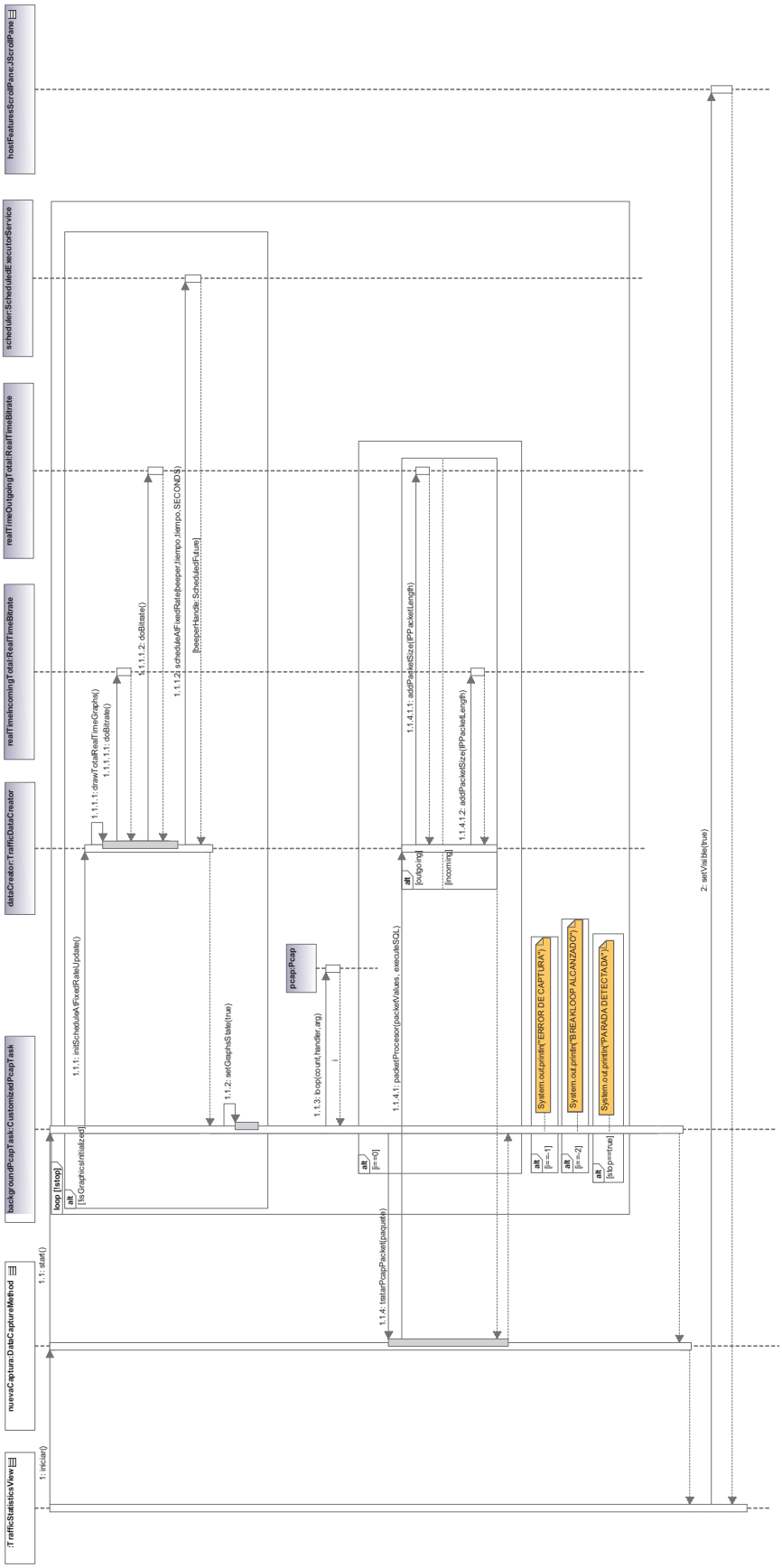


Gráfico 3.133 Diagrama de secuencia para explicar en detalle el método initDataCapture() de la clase TrafficStatisticsView destacando los mensajes que “realizan” los casos de uso citados al inicio

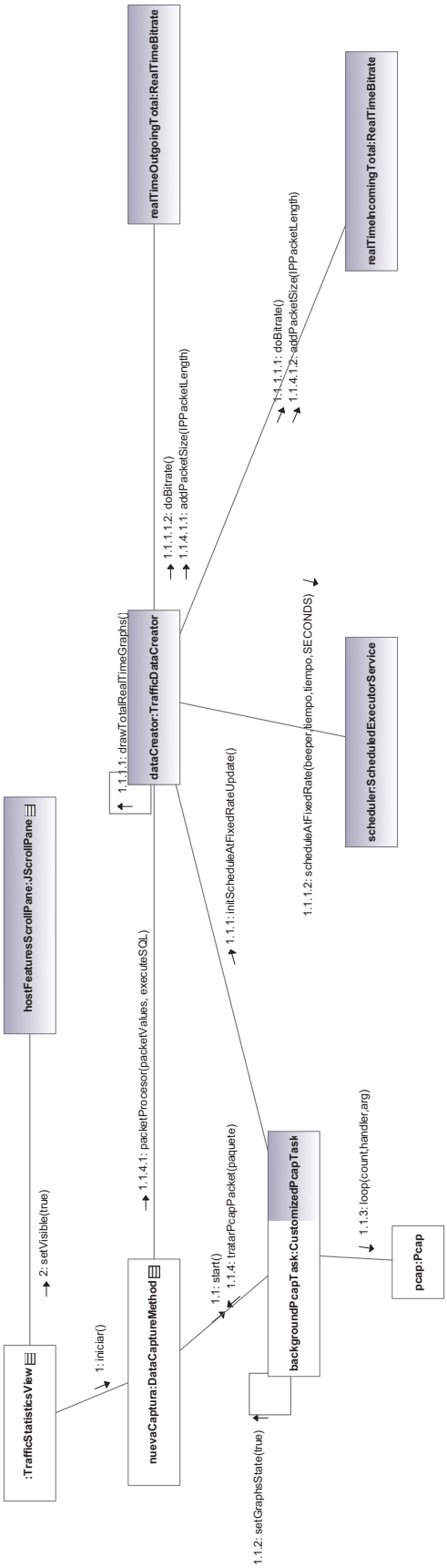


Gráfico 3.134 Diagrama de colaboración para explicar en detalle el método initDataCapture() de la clase TrafficStatisticsView destacando los mensajes que “realizan” los casos de uso citados al inicio

initDataCapture()	
Mensaje	Significado
1: iniciar()	Crea e inicializa un subproceso en segundo plano para la captura de datos monitoreados.
1.1: start()	Permite inicializar un manejador de tareas para la captura de un objeto Pcap. Este corre a nivel de bakground añadiendo métodos de control y métodos de estado para loop de captura.
1.1.1: initScheduleAtFixedRateUpdate()	Actualiza la base de datos con la lista de hosts existentes obtenida del monitoreo de tráfico Internet.
1.1.1.1: drawTotalRealTimeGraphs()	Gráfica el tráfico incoming y outgoing en tiempo real.
1.1.1.1.1: doBitrate() 1.1.1.1.2: doBitrate()	Asigna un hilo de ejecución para la actualización de la representación gráfica con los valores calculados periódicamente de manera automática.
1.1.1.2: scheduleAtFixedRate(beeper,tiempo,tiempo,SECONDS)	Permite actualizar la base de datos con los valores monitoreados cada 20 segundos.
1.1.2: setGraphsState(true)	Asigna el estado de la bandera para indicar que los gráficos están inicializados o no.
1.1.3: loop(count,handler,arg)	Permite capturar un paquete a la vez dentro de un loop que se ejecuta cada segundo y verifica si el paquete monitoreado no tiene errores.
1.1.4: tratarPcapPacket(paquete)	Decodifica y trata el paquete cuando este llega desde la interfaz de red.
1.1.4.1: packetProcesor(packetValues, executeSQL)	Procesa los paquetes de la interfaz de red, y clasifica la información agregando nuevos hosts, actualizando los valores de tráfico entrante y saliente.
1.1.4.1.1: addPacketSize(IPPacketLength) 1.1.4.1.2: addPacketSize(IPPacketLength)	Añade la longitud de los paquetes capturados en tiempo real a una lista.
2: setVisible(true)	Hace visible la lista de hosts.

Tabla 3.16 Mensajes del diagrama de colaboración del gráfico 3.132

Casos de uso:

- **Desplegar un sniffer y guardar contenido <<include>> Capturar tráfico de Internet**

- **Graficar tráfico en tiempo real por hosts <<include>> Capturar tráfico de Internet**

El usuario genera un evento sobre el botón de inicio del sniffer una vez seleccionadas las estaciones de trabajo y los filtros de protocolos y puertos. Ese evento es capturado por la aplicación y genera la siguiente secuencia de mensajes contenidas en el gráfico 3.135.

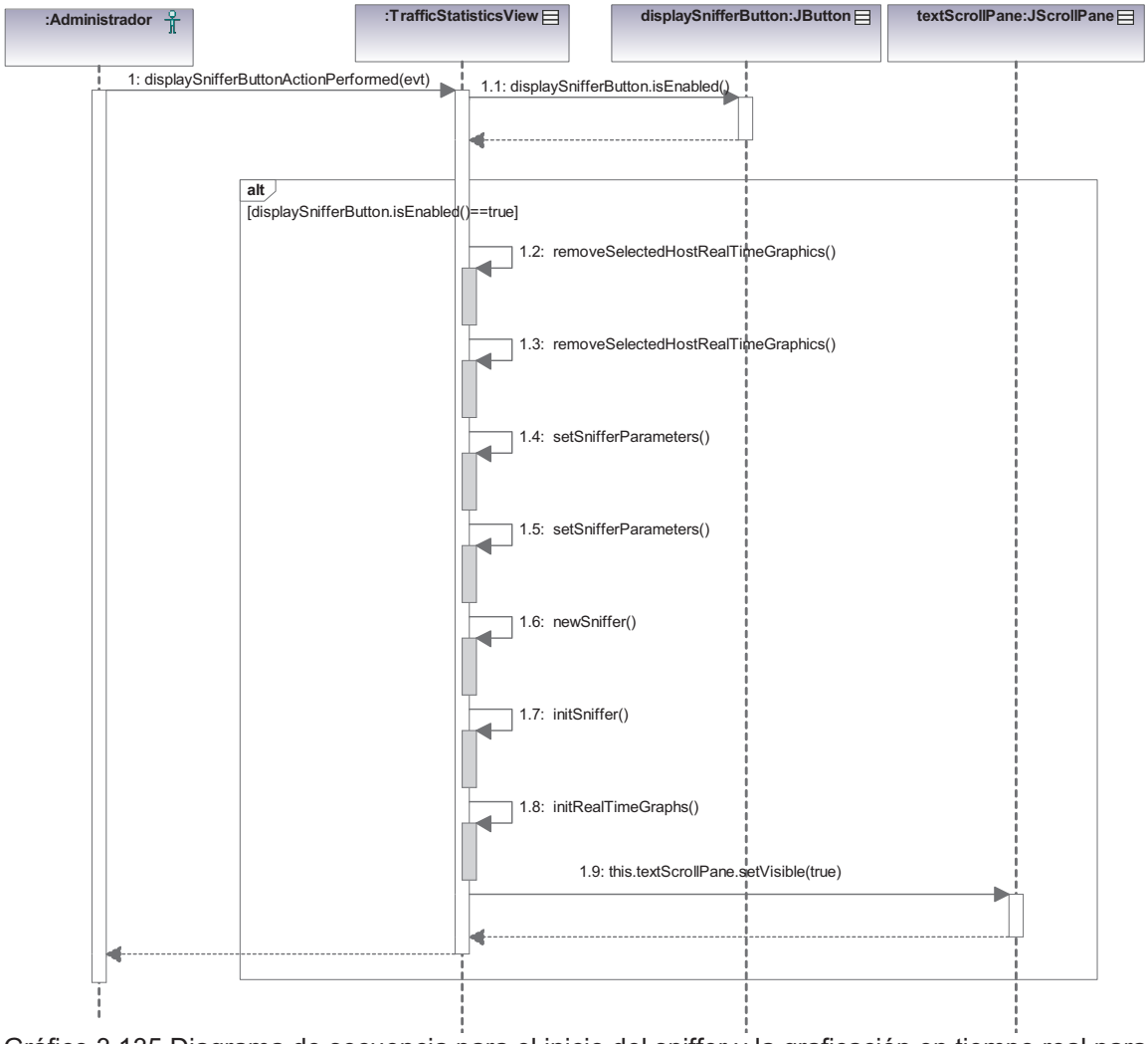


Gráfico 3.135 Diagrama de secuencia para el inicio del sniffer y la graficación en tiempo real para las estaciones de trabajo seleccionadas del método displaySnifferButtonActionPerformed(…) de la clase TrafficStatisticsView

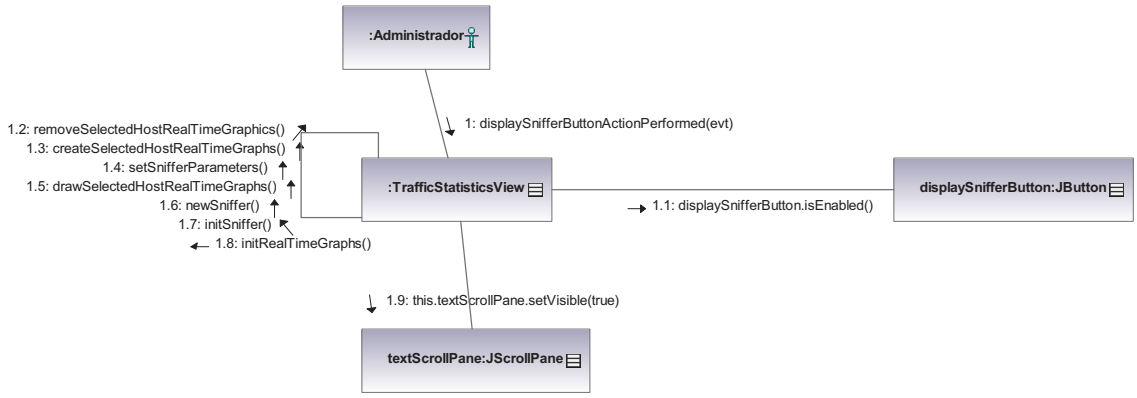


Gráfico 3.136 Diagrama de colaboración para el inicio del sniffer y la graficación en tiempo real para las estaciones de trabajo seleccionadas del método displaySnifferButtonActionPerformed(…) de la clase TrafficStatisticsView

displaySnifferButtonActionPerformed(…)	
Mensaje	Significado
1: displaySnifferButtonActionPerformed(evt)	Inicializa la funcionalidad del sniffer y la graficación de bitrate del tráfico de Internet por hosts seleccionados, con un evento generado por el usuario.
1.1: displaySnifferButton.isEnabled()	Verifica si el botón de inicio está habilitado para su uso.
1.2: removeSelectedHostRealTimeGraphics()	Remueve las formas dibujadas del tráfico de Internet de los hosts sobre los paneles y cancela la graficación automática.
1.3: createSelectedHostRealTimeGraphs()	Inicializa los objetos del tipo RealTimeBitrate que representan los gráficos de incoming y outgoing traffic en tiempo real de los hosts seleccionados.
1.4: setSnifferParameters()	Recupera los hosts seleccionados y los parámetros necesarios para la implementación del sniffer.
1.5: drawSelectedHostRealTimeGraphs()	Permite dibujar las diferentes formas que adopta el bitrate en un intervalo de tiempo cuando se monitorea el tráfico de Internet
1.6: newSniffer()	Instancia un nuevo sniffer para la captura de los datos y la representación gráfica del bitrate por hosts seleccionados.
1.7: initSniffer()	Asigna los estados de los botones cuando se inicia el sniffer presionando el botón en la barra principal de herramientas
1.8: initRealTimeGraphs()	Asigna los estados de los botones del gráfico de bitrate del tráfico en tiempo real por hosts cuando se inicia el sniffer presionando el botón en la barra de herramientas
1.9: textScrollPane.setVisible(true)	Hace visible al contenedor del área de texto que contendrá los paquetes decodificados.

Tabla 3.17 Mensajes del diagrama de colaboración del gráfico 3.136

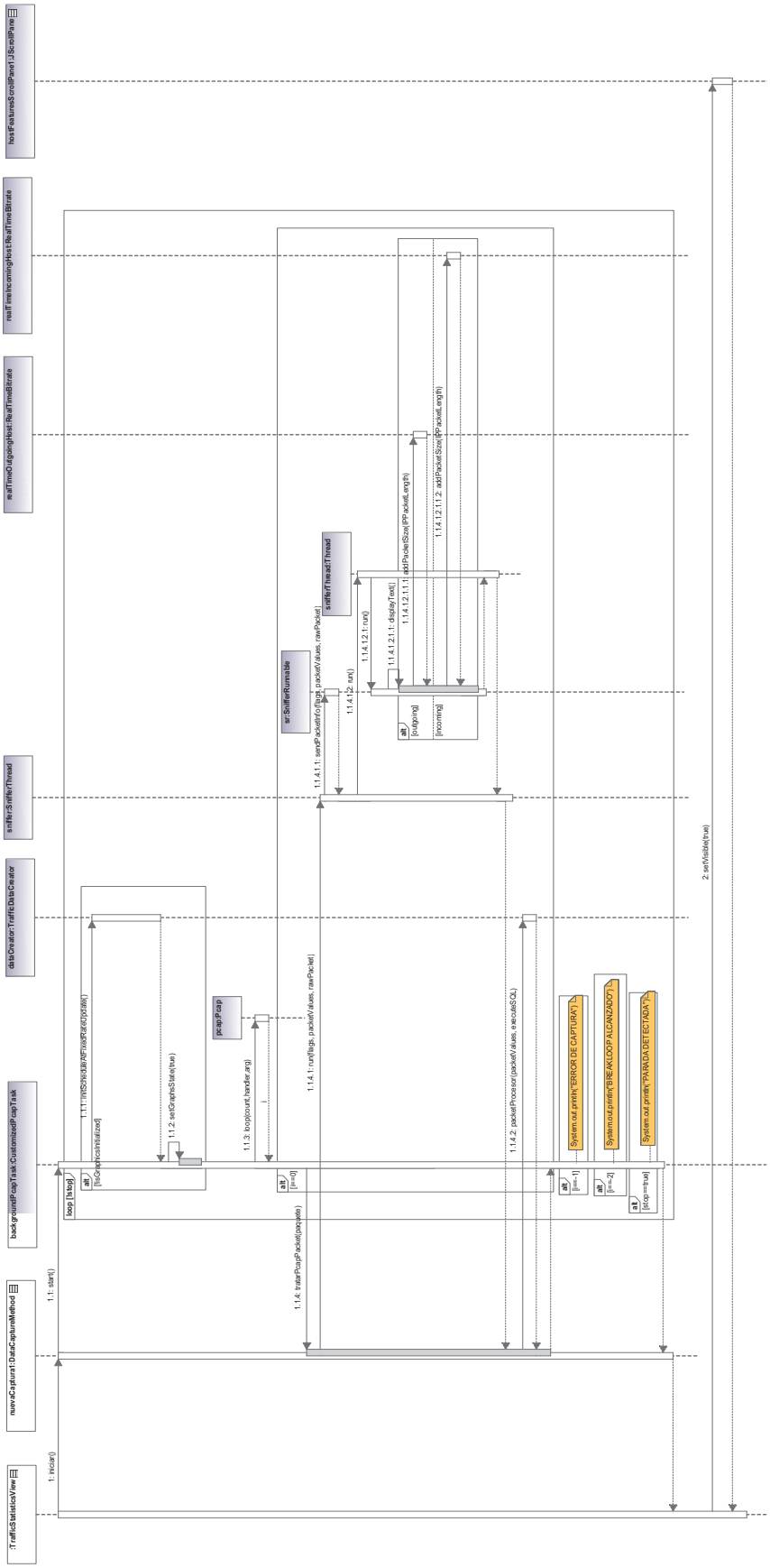


Gráfico 3.137 Diagrama de secuencia para explicar en detalle el método initDataCapture() de la clase TrafficStatisticsView destacando los mensajes que “realizan” los casos de uso para el sniffer y la graficación tiempo real por estaciones seleccionadas

El sniffer trabaja conjuntamente con la captura de paquetes. La secuencia de mensajes para este caso particular se detalla a continuación en el gráfico 3.138.

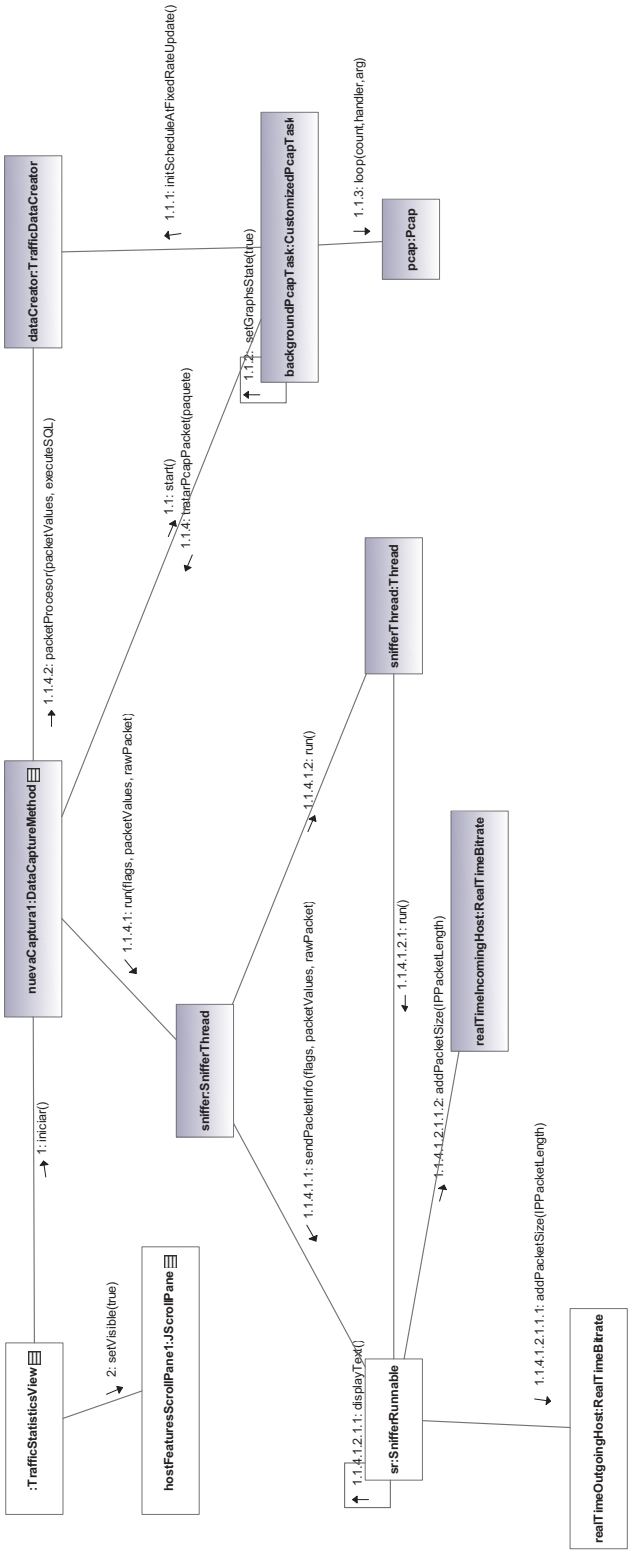


Gráfico 3.138 Diagrama de colaboración para explicar en detalle el método initDataCapture() de la clase TrafficStatisticsView destacando los mensajes que “realizan” los casos de uso para el sniffer y la graficación tiempo real por estaciones seleccionadas

initDataCapture()	
Mensaje	Significado
1: iniciar()	Crea e inicializa un subproceso en segundo plano para la captura de datos monitoreados.
1.1: start()	Permite inicializar un manejador de tareas para la captura de un objeto Pcap. Este corre a nivel de bakground añadiendo métodos de control y métodos de estado para loop de captura.
1.1.1: initScheduleAtFixedRateUpdate()	Actualiza la base de datos con la lista de hosts existentes obtenida del monitoreo de tráfico Internet.
1.1.2: setGraphsState(true)	Asigna el estado de la bandera para indicar que los gráficos están inicializados o no.
1.1.3: loop(count,handler,arg)	Permite capturar un paquete a la vez dentro de un loop que se ejecuta cada segundo y verifica si el paquete monitoreado no tiene errores.
1.1.4: tratarPcapPacket(paquete)	Decodifica y trata el paquete cuando este llega desde la interfaz de red.
1.1.4.1: run(flags, packetValues, rawPacket)	Crea un hilo que permite ejecutar el método run() de la clase SnifferRunnable y se ejecuta cada vez que se capture un nuevo paquete al dispositivo de red.
1.1.4.1.1: sendPacketInfo(flags, packetValues, rawPacket)	Asigna los nuevos valores del paquete decodificado y de las banderas que identifican si el paquete es TCP, UDP, ICMP.
1.1.4.1.2: run()	Ejecuta el hilo con los indicadores que identifican el tipo de protocolo y el paquete decodificado ingresados como parámetros.
1.1.4.1.2.1: run()	Ejecuta el método que permite desplegar los diferentes campos de los paquetes decodificados en el área de texto.
1.1.4.1.2.1.1: displayText()	Despliega los diferentes campos de los paquetes decodificados en el área de texto.
1.1.4.1.2.1.1.1: addPacketSize(IPPacketLength) 1.1.4.1.2.1.1.2: addPacketSize(IPPacketLength)	Añade la longitud de los paquetes capturados en tiempo real a una lista.
1.1.4.2: packetProcesor(packetValues, executeSQL)	Procesa los paquetes de la interfaz de red, y clasifica la información agregando nuevos hosts, actualizando los valores de tráfico entrante y saliente.
2: setVisible(true)	Hace visible la lista de hosts.

Tabla 3.18 Mensajes del diagrama de colaboración del gráfico 3.138

Caso de uso:

- **Diferenciar tráfico (host, protocolos, puertos)**

La diferenciación de tráfico se establece por medio de los filtros del sniffer que se asignan con el método setSnifferParameters().

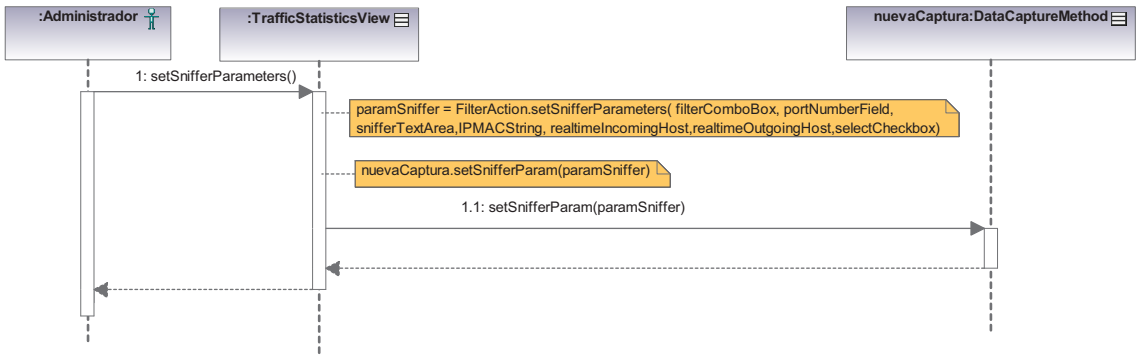


Gráfico 3.139 Diagrama de secuencia para la selección de los parámetros necesarios en la diferenciación de tráfico del sniffer del método setSnifferParameters() de la clase TrafficStatisticsView

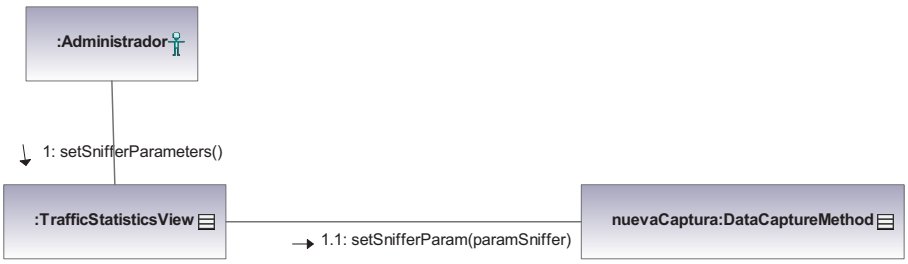


Gráfico 3.140 Diagrama de colaboración para la selección de los parámetros necesarios en la diferenciación de tráfico del sniffer del método setSnifferParameters() de la clase TrafficStatisticsView

setSnifferParameters()	
Mensaje	Significado
1: setSnifferParameters()	Recupera los hosts seleccionados y los parámetros necesarios para la implementación del sniffer.
1.1: setSnifferParam(paramSniffer)	Asigna los parámetros del tipo SnifferParameters requeridos para instanciar un sniffer que permita controlar los gráficos de bitrate en tiempo real por hosts, así como la impresión de los paquetes decodificados y toda la información relevante como filtro pertenecientes a ciertos host seleccionados por el usuario.

Tabla 3.19 Mensajes del diagrama de colaboración del gráfico 3.140

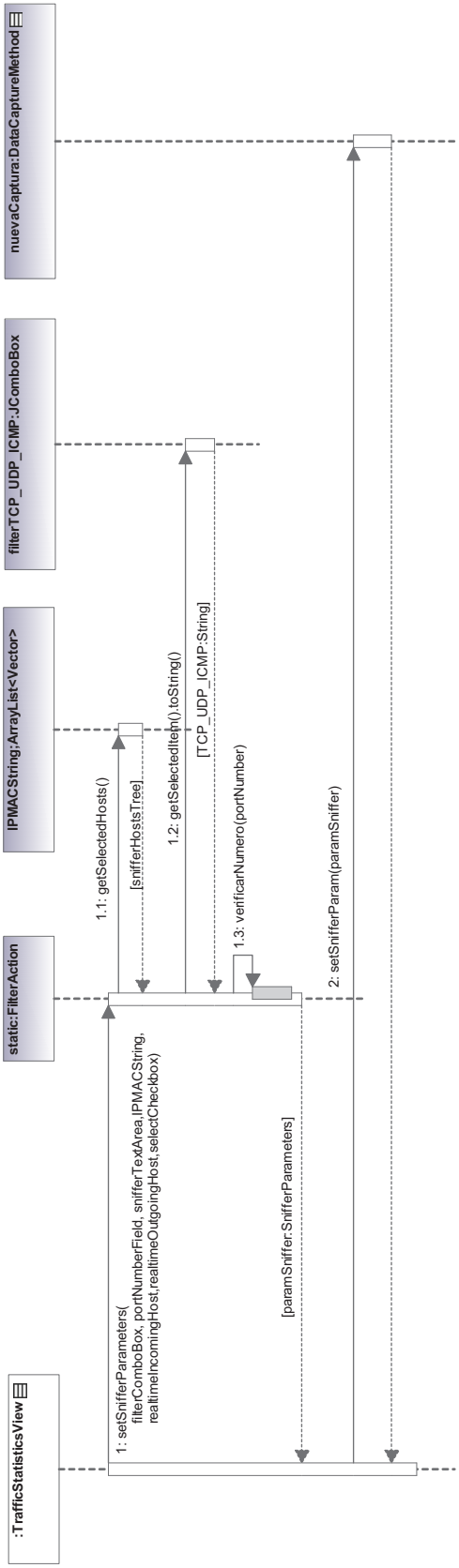


Gráfico 3.141 Diagrama de secuencia que explica el método estático setSnifferParameters() de la clase FilterAction a detalle para el cumplimiento del caso de uso de diferenciación de tráfico

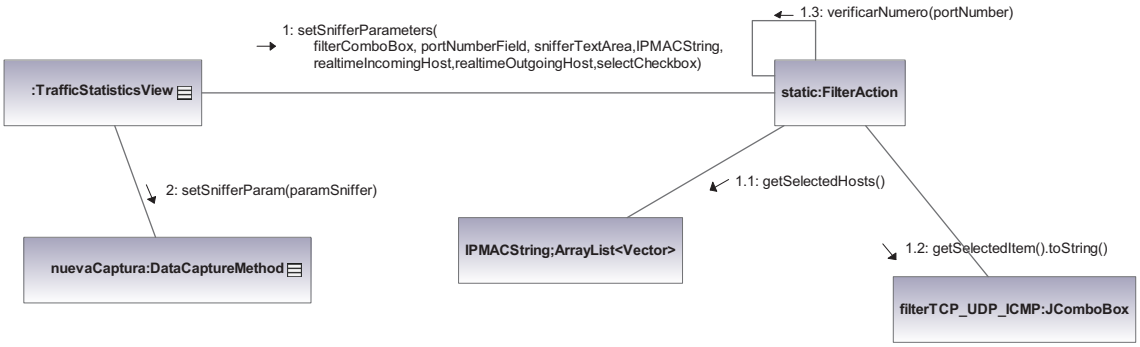


Gráfico 3.142 Diagrama de colaboración que explica el método estático setSnifferParameters() de la clase FilterAction a detalle para el cumplimiento del caso de uso de diferenciación de tráfico

setSnifferParameters()	
Mensaje	Significado
1: setSnifferParameters(filterComboBox, portNumberField, snifferTextArea, IPMACString, realtimeIncomingHost, realtimeOutgoingHost,selectCheckbox)	Recupera los hosts seleccionados y los parámetros necesarios para la implementación del sniffer.
1.1: getSelectedHosts()	Devuelve el host seleccionado dentro un ArrayList que contiene una relación unívoca entre la dirección MAC e IP de los hosts monitoreados.
1.2: getSelectedItem().toString()	Devuelve el item seleccionado dentro de un JComboBox que contiene los protocolos TCP, UDP, ICMP para el correspondiente filtro.
1.3: verificarNumero(portNumber)	Verifica la validez del entero ingresado por el usuario que representa el número de puerto indicando que el valor -1 significa un filtro con todos los puertos a escuchar.
2: setSnifferParam(paramSniffer)	Asigna los parámetros del tipo SnifferParameters requeridos para instanciar un sniffer que permita controlar los gráficos de bitrate en tiempo real por hosts, así como la impresión de los paquetes decodificados y toda la información relevante como filtro pertenecientes a ciertos hosts seleccionados por el usuario.

Tabla 3.20 Mensajes del diagrama de colaboración del gráfico 3.142

Caso de uso:

- **Almacenar y recuperar de disco**

Se lo hace de manera automática para almacenar los datos de la captura en la base de datos.

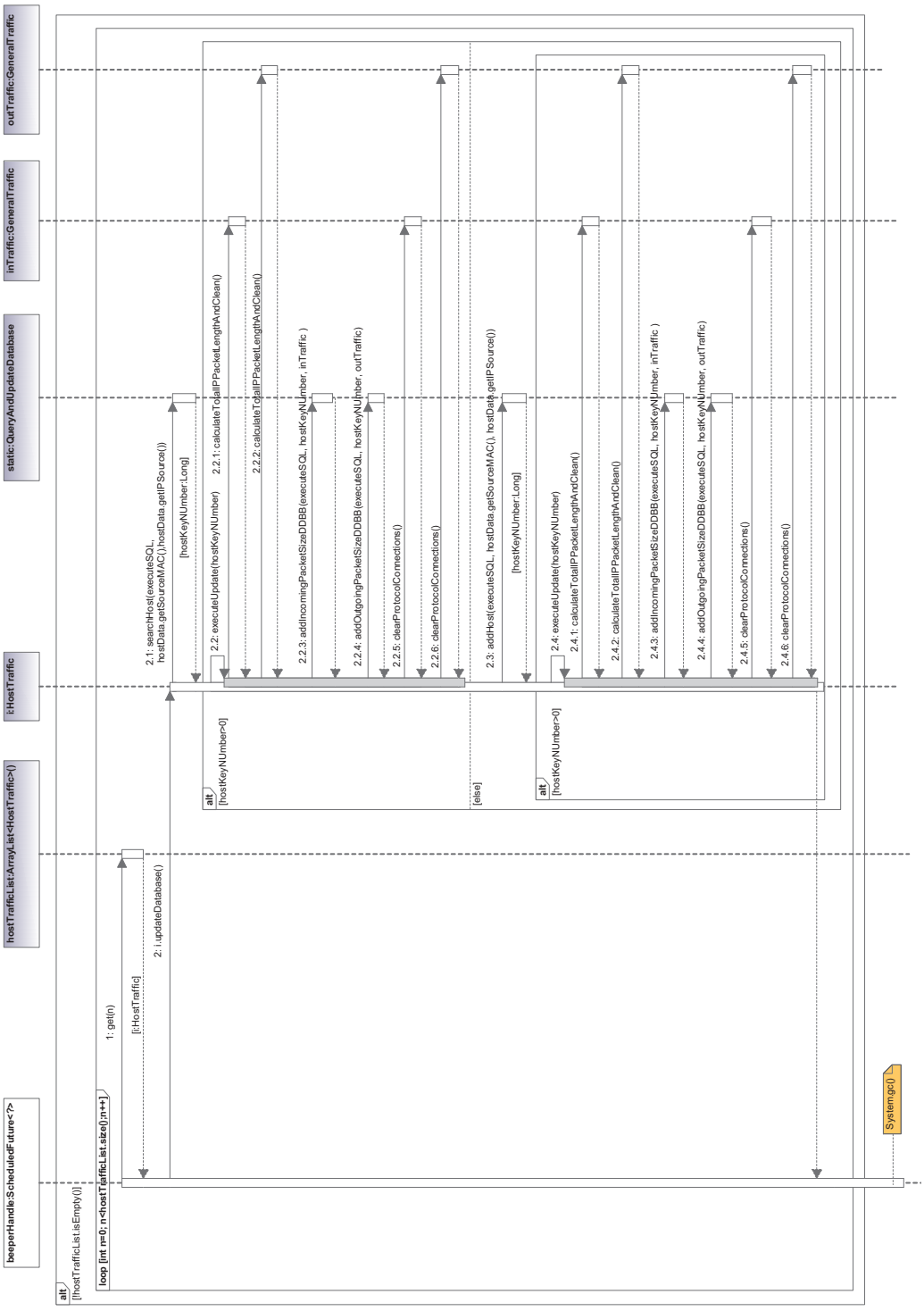


Gráfico 3.143 Diagrama de secuencia del método run() de un objeto Runnable que es controlado por un objeto ScheduledFuture para la actualización periódica de la base de datos cada 20 segundos

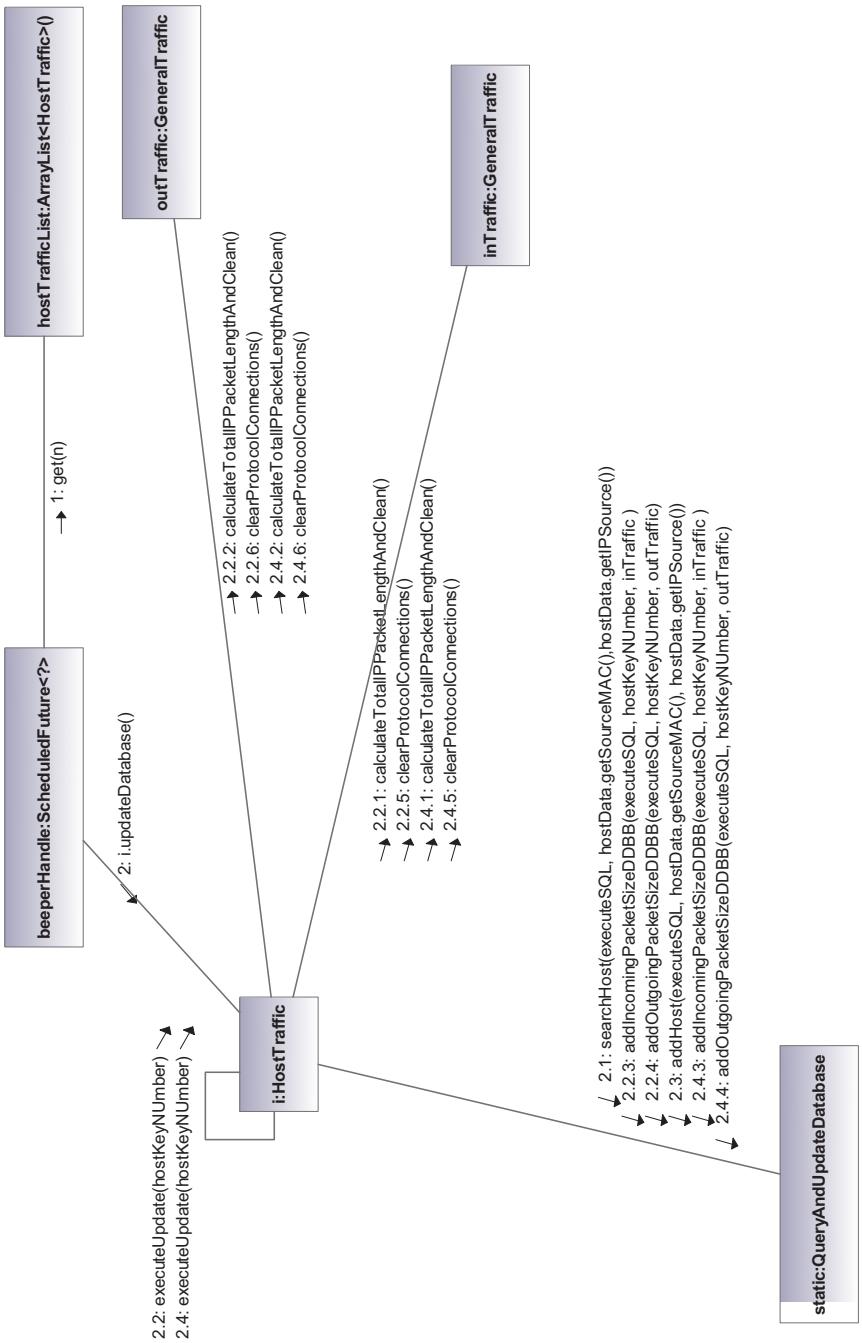


Gráfico 3.144 Diagrama de colaboración del método run() de un objeto Runnable que es controlado por un objeto ScheduledFuture para la actualización periódica de la base de datos cada 20 segundos

ScheduledFuture	
Mensaje	Significado
1: get(n)	Recupera una lista de hosts existentes obtenida del monitoreo de tráfico Internet.
2: i.updateDatabase()	Obtiene identificador del host dentro de la base de datos y ejecuta la actualización del tráfico asociado a esta estación de trabajo.

2.1: searchHost(executeSQL, hostData.getSourceMAC(),hostData.getIPSource())	Ejecuta la consulta MySQL para la búsqueda de una estación de trabajo por MacID e IP.
2.2: executeUpdate(hostKeyNUmber) 2.4: executeUpdate(hostKeyNUmber)	Actualiza la base de datos con la información del host correspondiente al identificador hostKeyNUmber del tipo Long.
2.3: addHost(executeSQL, hostData.getSourceMAC(), hostData.getIPSource())	Retorna el identificador del último host luego de añadirlo.
2.2.1: calculateTotalIPPacketLengthAndClean() 2.2.2: calculateTotalIPPacketLengthAndClean() 2.4.1: calculateTotalIPPacketLengthAndClean() 2.4.2: calculateTotalIPPacketLengthAndClean()	Calcula el tamaño acumulado de los valores de tamaño almacenados en el ArrayList protocols y limpia la respectiva lista de cada ProtocolTrafficByConnection existente en la lista protocols.
2.2.3: addIncomingPacketSizeDDBB(executeSQL, hostKeyNUmber, inTraffic) 2.4.3: addIncomingPacketSizeDDBB(executeSQL, hostKeyNUmber, inTraffic)	Actualiza la base de datos con la información incoming del host correspondiente al identificador hostKeyNUmber del tipo Long para el tráfico entrante.
2.2.4: addOutgoingPacketSizeDDBB(executeSQL, hostKeyNUmber, outTraffic) 2.4.4: addOutgoingPacketSizeDDBB(executeSQL, hostKeyNUmber, outTraffic)	Actualiza la base de datos con la información outgoing del host correspondiente al identificador hostKeyNUmber del tipo Long para el tráfico entrante.
2.2.5: clearProtocolConnections() 2.2.6: clearProtocolConnections() 2.4.5: clearProtocolConnections() 2.4.6: clearProtocolConnections()	Limpia ArrayList de las conexiones almacenadas.

Tabla 3.21 Mensajes del diagrama de colaboración del gráfico 3.144

3.8.2. QUERY STATISTICS

Caso de uso:

- **Mostrar y guardar gráfico de análisis estadístico**

El usuario inicia la conexión con la base de datos y aparece una lista con las opciones de generación de gráficos. Para acceder a ellos el usuario debe realizar doble clic sobre la opción deseada. Este evento es capturado por la aplicación como se describe en el gráfico 3.145a y 3.145b.

Gráfico 3.145a Diagrama de secuencia para la selección del gráfico deseado mediante un doble clic sobre la lista desplegada (método doubleClick(..) de la clase StatisticsGraphicsTypesJList)

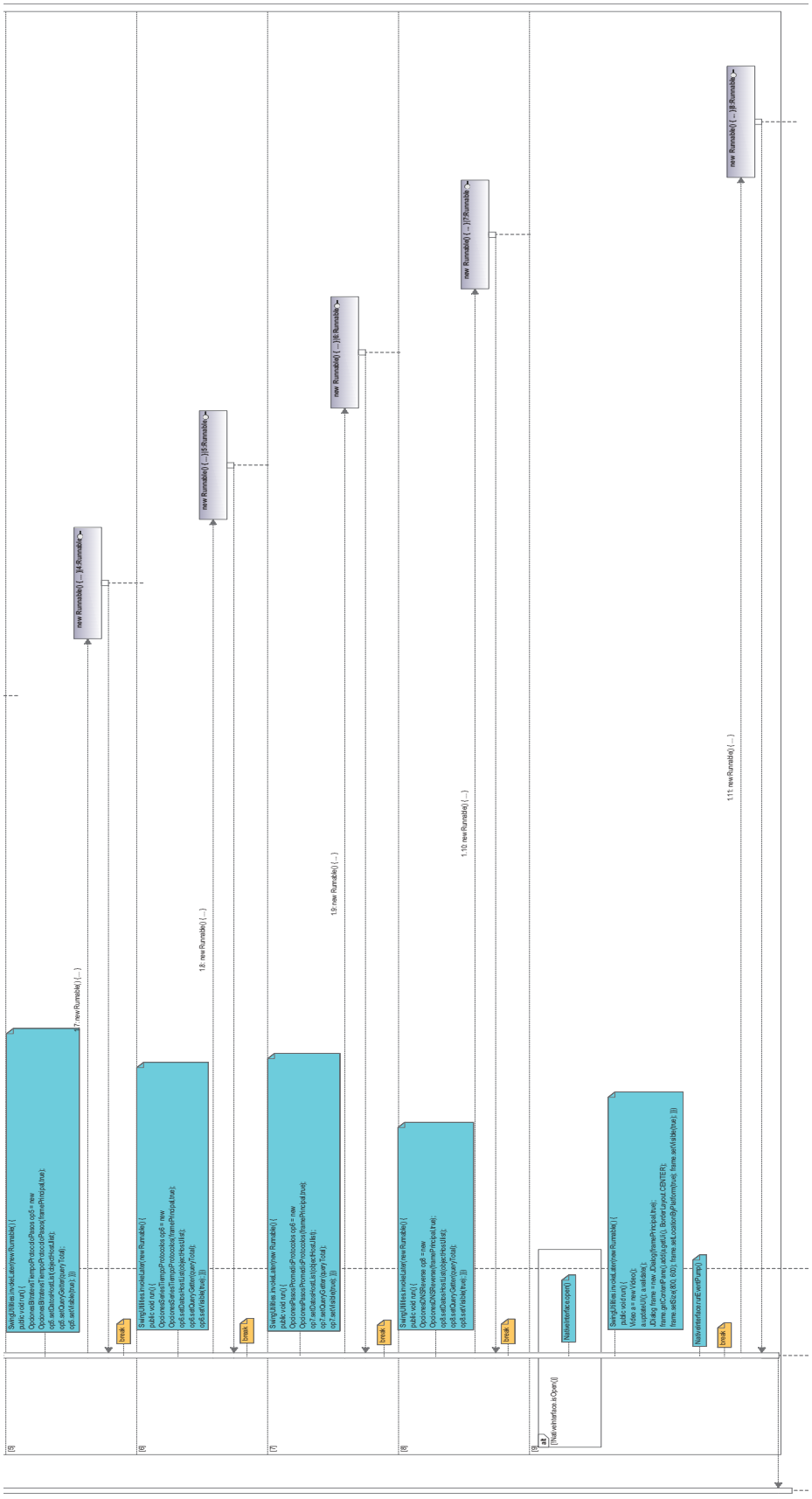


Gráfico 3.145b Diagrama de secuencia para la selección del gráfico deseado mediante un doble clic sobre la lista desplegada (método doubleClick(..) de la clase StatisticsGraphicsTypesJList)

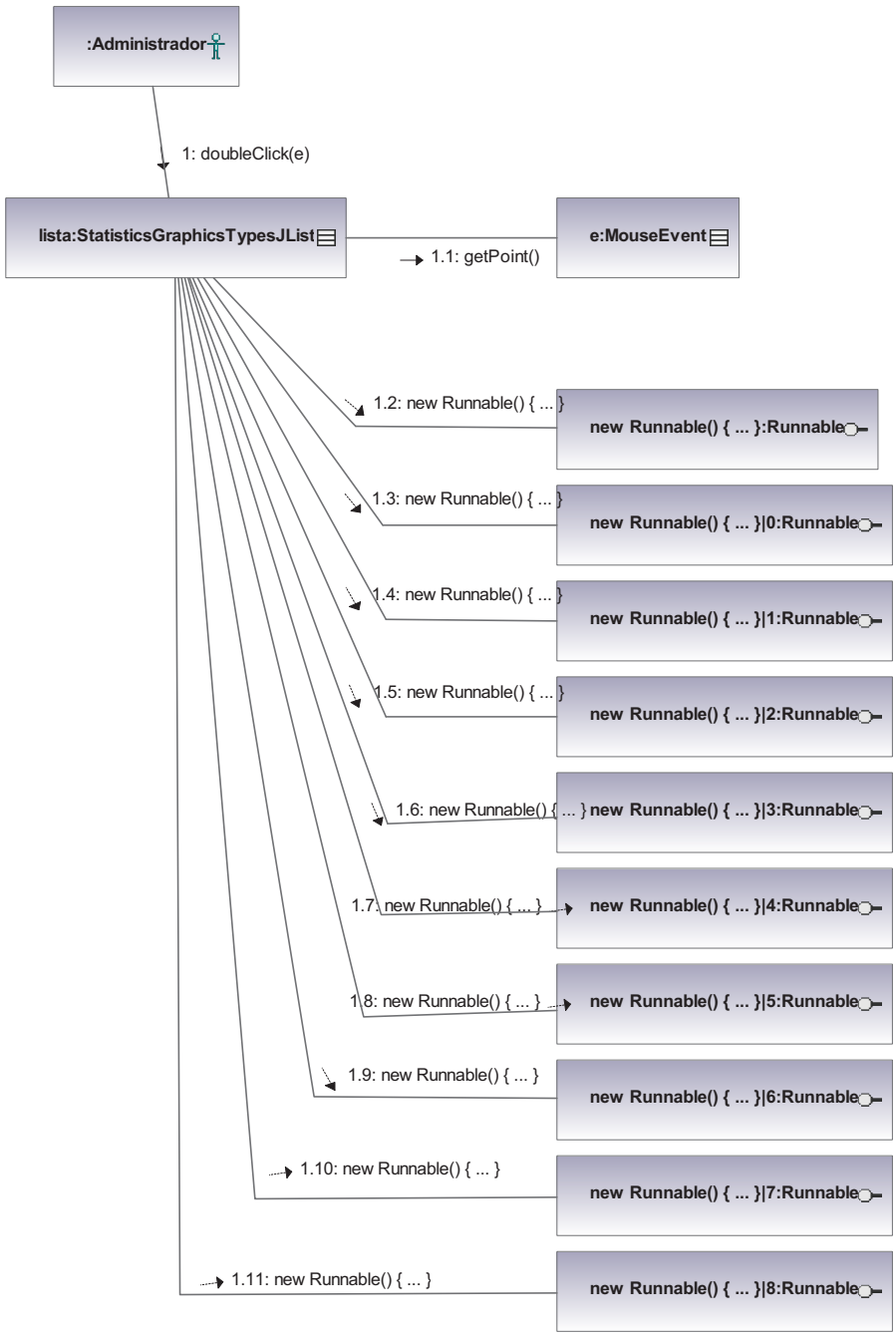


Gráfico 3.146 Diagrama de colaboración para la selección del gráfico deseado mediante un doble clic sobre la lista desplegada (método doubleClick(..) de la clase StatisticsGraphicsTypesJList)

doubleClick(…)	
Mensaje	Significado
1: doubleClick(e)	Instancia una ventana de diálogo por cada uno de los diagramas estadísticos para su posterior representación gráfica. Se instancia dicha ventana de diálogo a través de un evento del mouse que determina qué tipo de gráfico el usuario desea generar.

1.1: getPoint()	Retorna el índice de la opción elegida dentro de la lista de diagramas estadísticos.
1.2: new Runnable() { ... } 1.3: new Runnable() { ... } 1.4: new Runnable() { ... } 1.5: new Runnable() { ... } 1.6: new Runnable() { ... } 1.7: new Runnable() { ... } 1.8: new Runnable() { ... } 1.9: new Runnable() { ... } 1.10: new Runnable() { ... } 1.11:new Runnable() { ... }	Instancia un hilo de ejecución para inicializar una ventana de diálogo y la generación de los diagramas estadísticos.

Tabla 3.22 Mensajes del diagrama de colaboración del gráfico 3.146

Cada diálogo implementa de manera particular el método drawChart(…) para cada tipo de gráfico como se ve a continuación. Si bien es cierto que la secuencia de mensajes es similar, la implementación de los métodos es diferente.

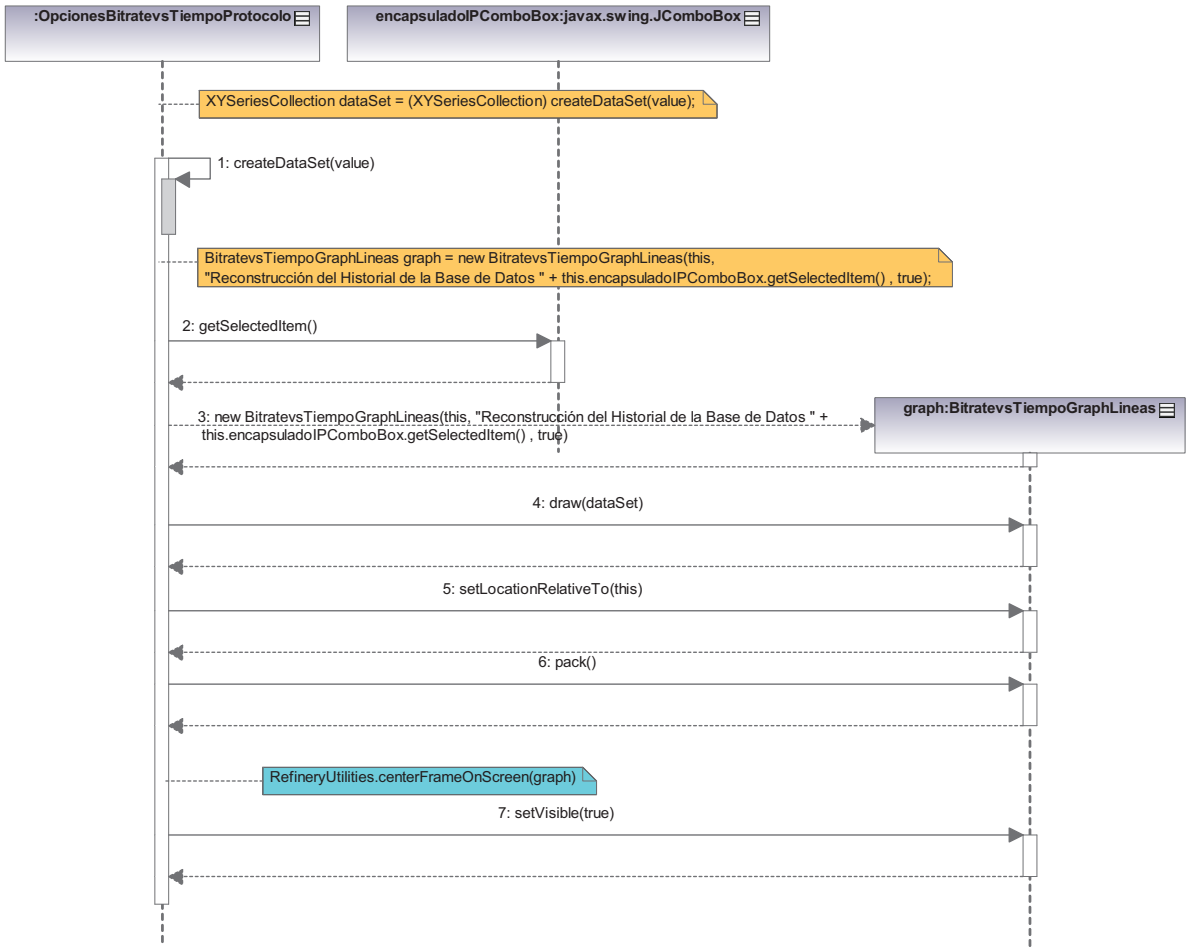


Gráfico 3.147 Diagrama de secuencia para la reconstrucción gráfica de los datos de la base.

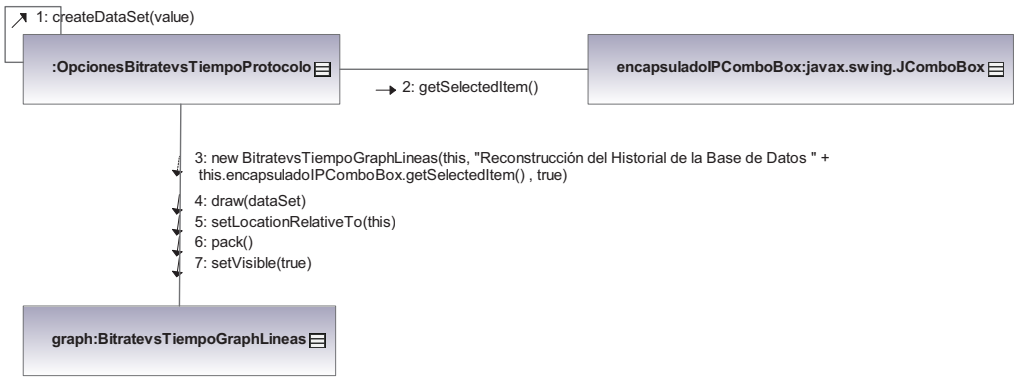


Gráfico 3.148 Diagrama de colaboración para la reconstrucción gráfica de los datos de la base.

OpcionesBitratevsTiempoProtocolo	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Reconstrucción del Historial de la Base de Datos Lineas.
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.
3: new BitratevsTiempoGraphLineas(this, "Reconstrucción del Historial de la Base de Datos " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de Reconstrucción del Historial de la Base de Datos Lineas para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección.
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: pack()	Organiza y empaqueta los componentes gráficos para su representación.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.23 Mensajes del diagrama de colaboración del gráfico 3.148

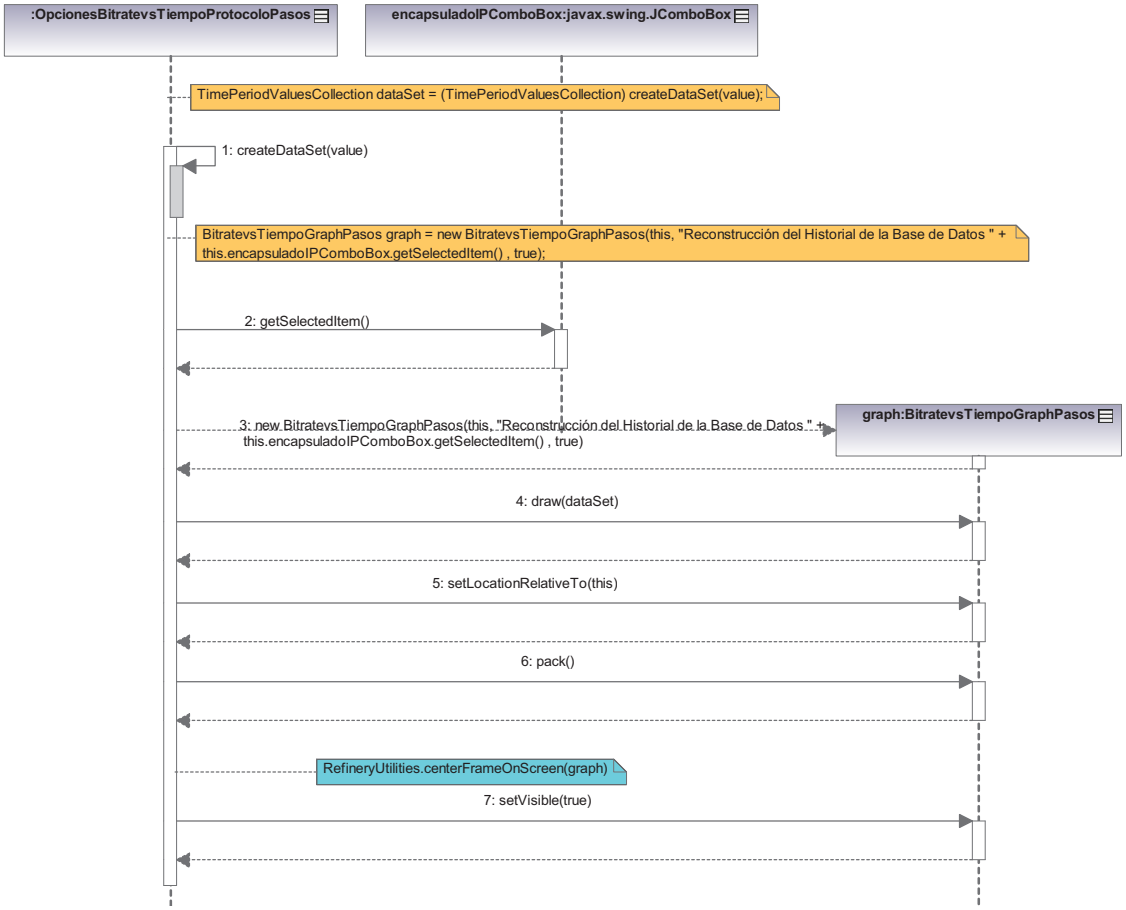


Gráfico 3.149 Diagrama de secuencia para la reconstrucción gráfica de los datos de la base usando pasos.

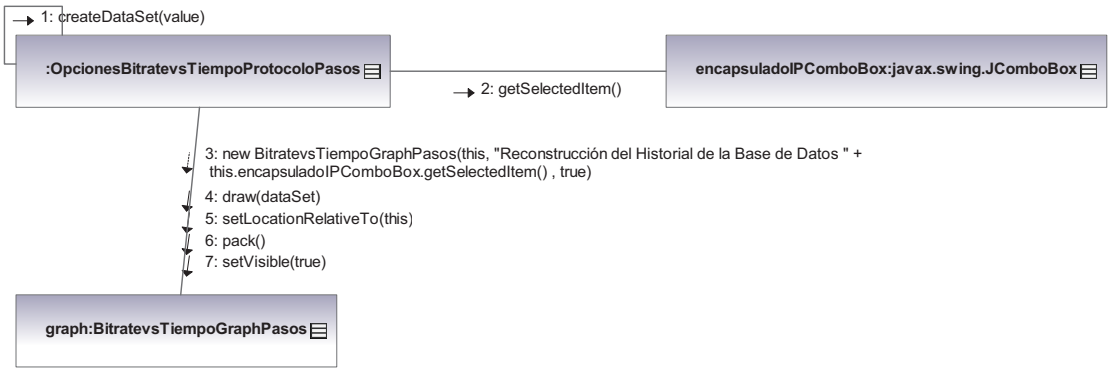


Gráfico 3.150 Diagrama de colaboración para la reconstrucción gráfica de los datos de la base usando pasos.

OpcionesBitratevsTiempoProtocoloPasos	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Reconstrucción del Historial de la Base de Datos Pasos.
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.

3: new BitratevsTiempoGraphPasos(this, "Reconstrucción del Historial de la Base de Datos " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de Reconstrucción del Historial de la Base de Datos Pasos para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: pack()	Organiza y empaqueta los componentes gráficos para su representación.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.24 Mensajes del diagrama de colaboración del gráfico 3.150

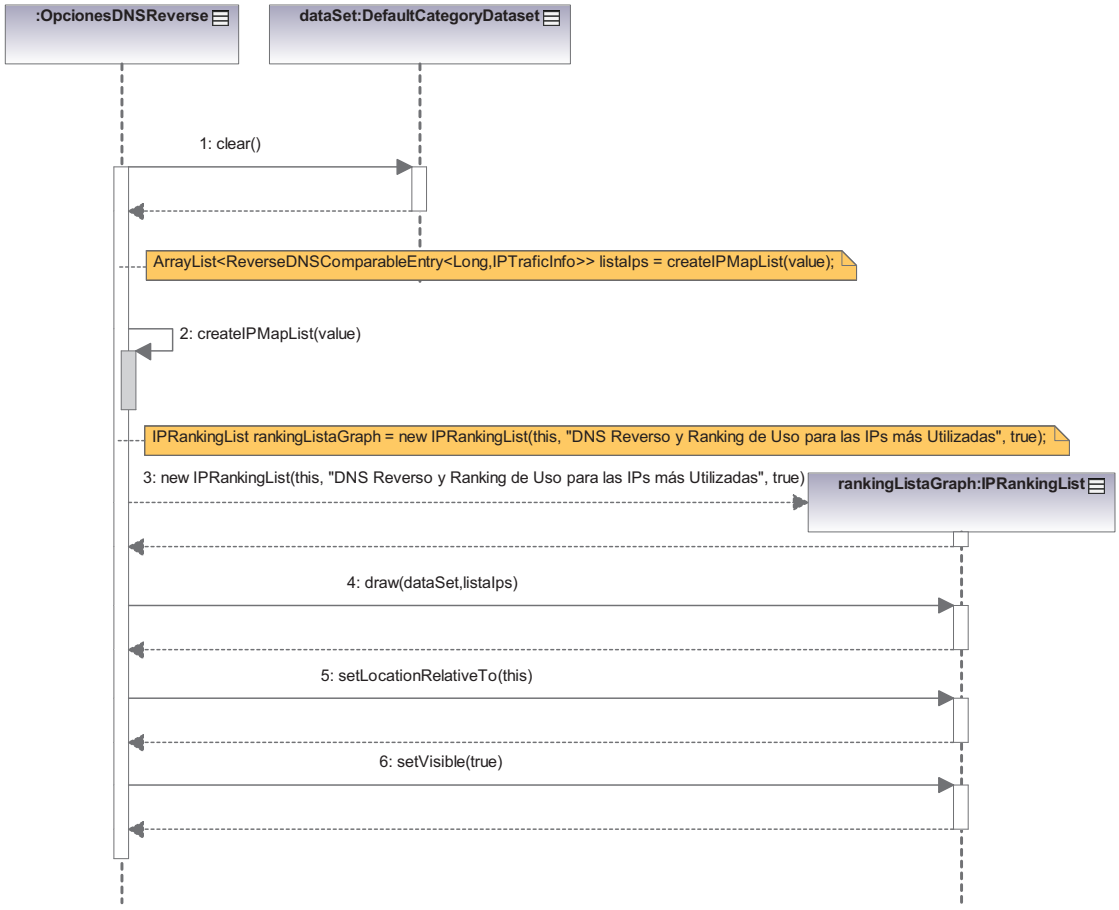


Gráfico 3.151 Diagrama de secuencia para el ranking de las IPs más utilizadas y la resolución inversa de nombres.

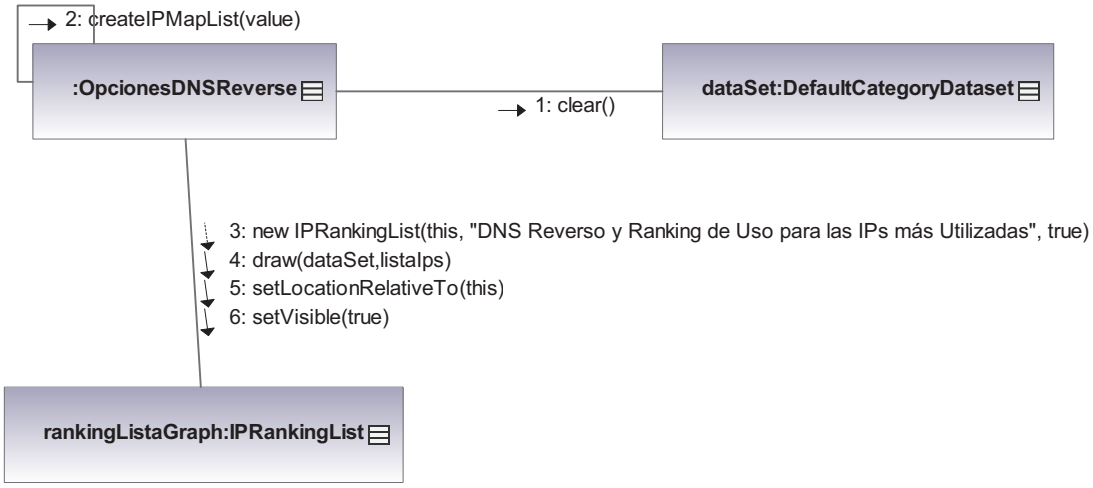


Gráfico 3.152 Diagrama de colaboración para el ranking de las IPs más utilizadas y la resolución inversa de nombres.

OpcionesDNSReverse	
Mensaje	Significado
1: clear()	Limpia todos los valores de la estructura de datos de un diagrama anterior para evitar inconsistencias.
2: createIPMapList(value)	Permite generar una lista de direcciones IP desde una ResultSet obtenido de una consulta a la base de datos pasando por un filtro seteado por el usuario en el cuadro de diálogo, por hosts, protocolo o puerto para la resolución de nombres de dominio.
3: new IPRankingList(this, "DNS Reverso y Ranking de Uso para las IPs más Utilizadas", true)	Instancia un objeto que permite realizar DNS Reverso y Ranking de Uso para las IPs más Utilizadas,el cual contiene tablas de las direcciones IPs más usadas así como de un gráfico en barras horizontales que representa la cantidad de flujo de datos para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.25 Mensajes del diagrama de colaboración del gráfico 3.152

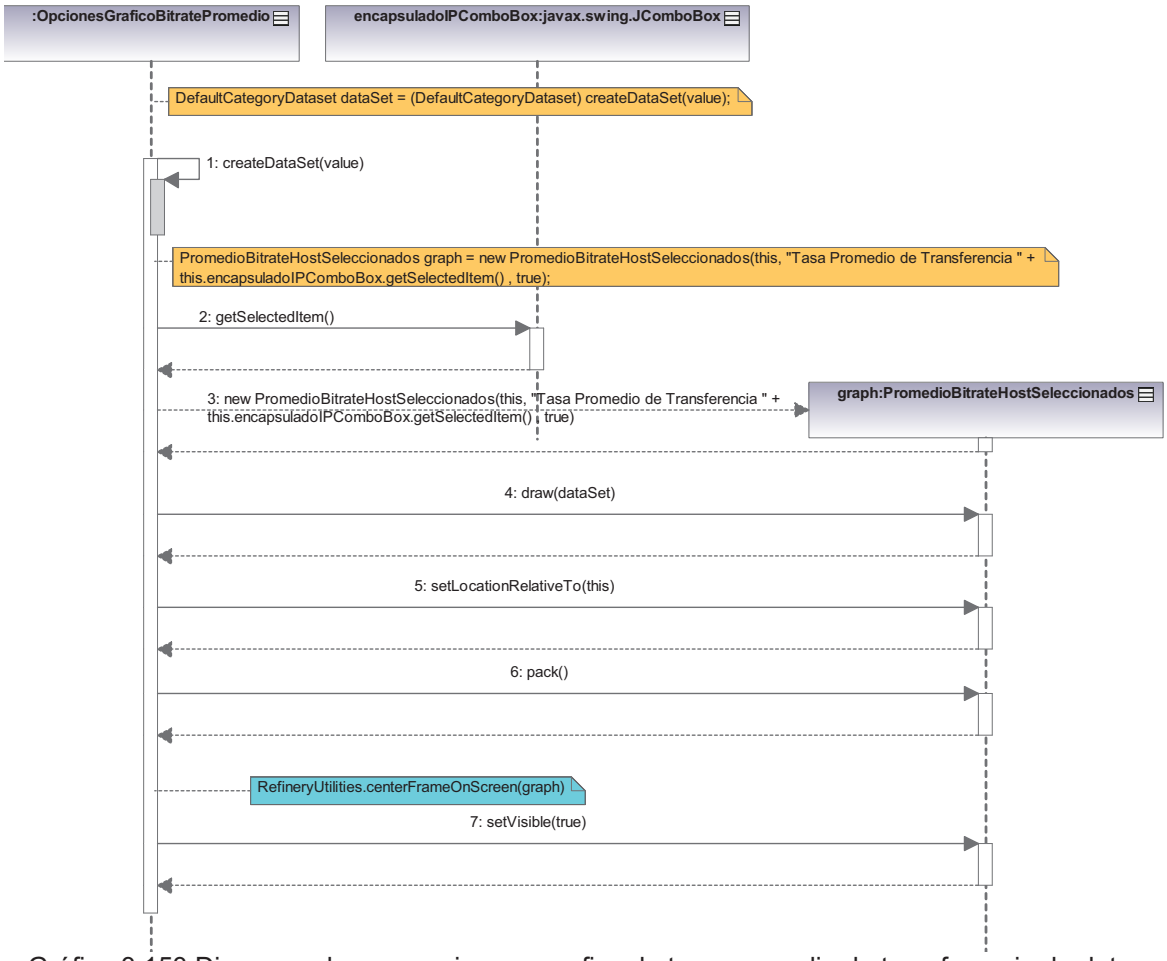


Gráfico 3.153 Diagrama de secuencia para graficar la tasa promedio de transferencia de datos

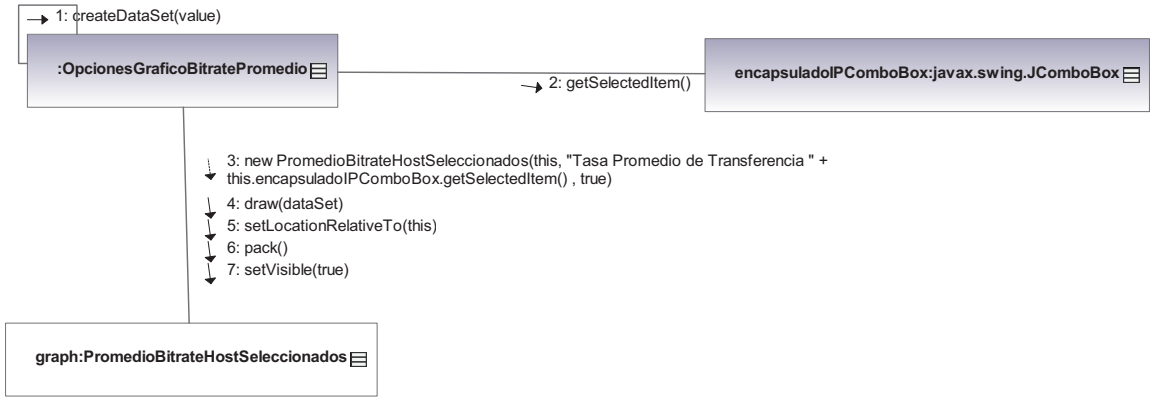


Gráfico 3.154 Diagrama de colaboración para graficar la tasa promedio de transferencia de datos

OpcionesGraficoBitratePromedio	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Tasa Promedio de Transferencia.
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.

3: new PromedioBitrateHostSeleccionados(this, "Tasa Promedio de Transferencia " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de Tasa Promedio de Transferencia para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: pack()	Organiza y empaqueta los componentes gráficos para su representación.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.26 Mensajes del diagrama de colaboración del gráfico 3.154

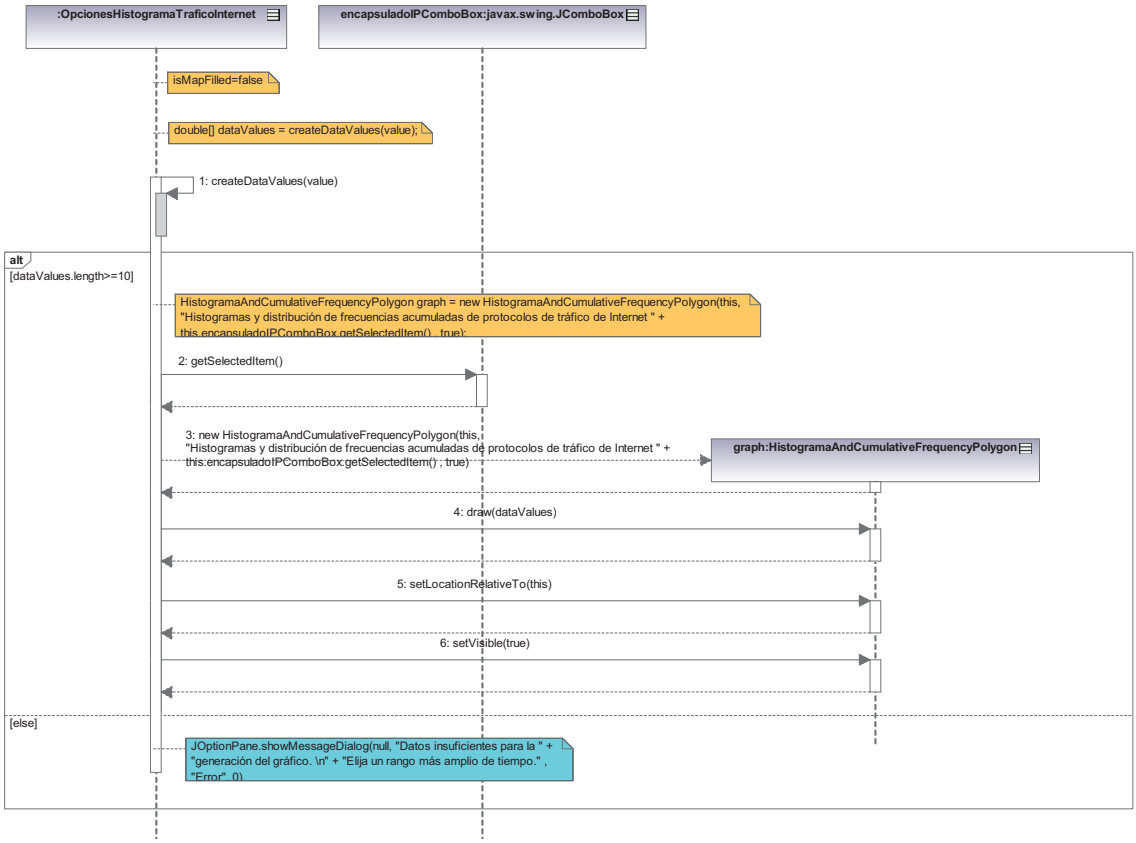


Gráfico 3.155 Diagrama de secuencia para obtener un histograma de frecuencias, frecuencias acumuladas y resumen de estadística de descriptiva de los valores de bitrate calculados

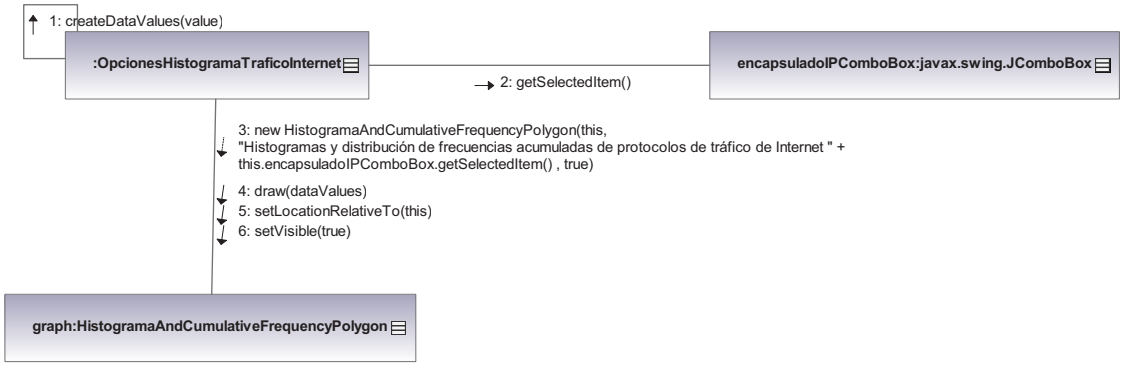


Gráfico 3.156 Diagrama de colaboración para obtener un histograma de frecuencias, frecuencias acumuladas y resumen de estadística de descriptiva de los valores de bitrate calculados

OpcionesHistogramaTraficoInternet	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Histogramas y distribución de frecuencias acumuladas de protocolos de tráfico de Internet
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.
3: new HistogramaAndCumulativeFrequencyPolygon(this, "Histogramas y distribución de frecuencias acumuladas de protocolos de tráfico de Internet " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de Histogramas y distribución de frecuencias acumuladas de protocolos de tráfico de Internet para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.27 Mensajes del diagrama de colaboración del gráfico 3.156

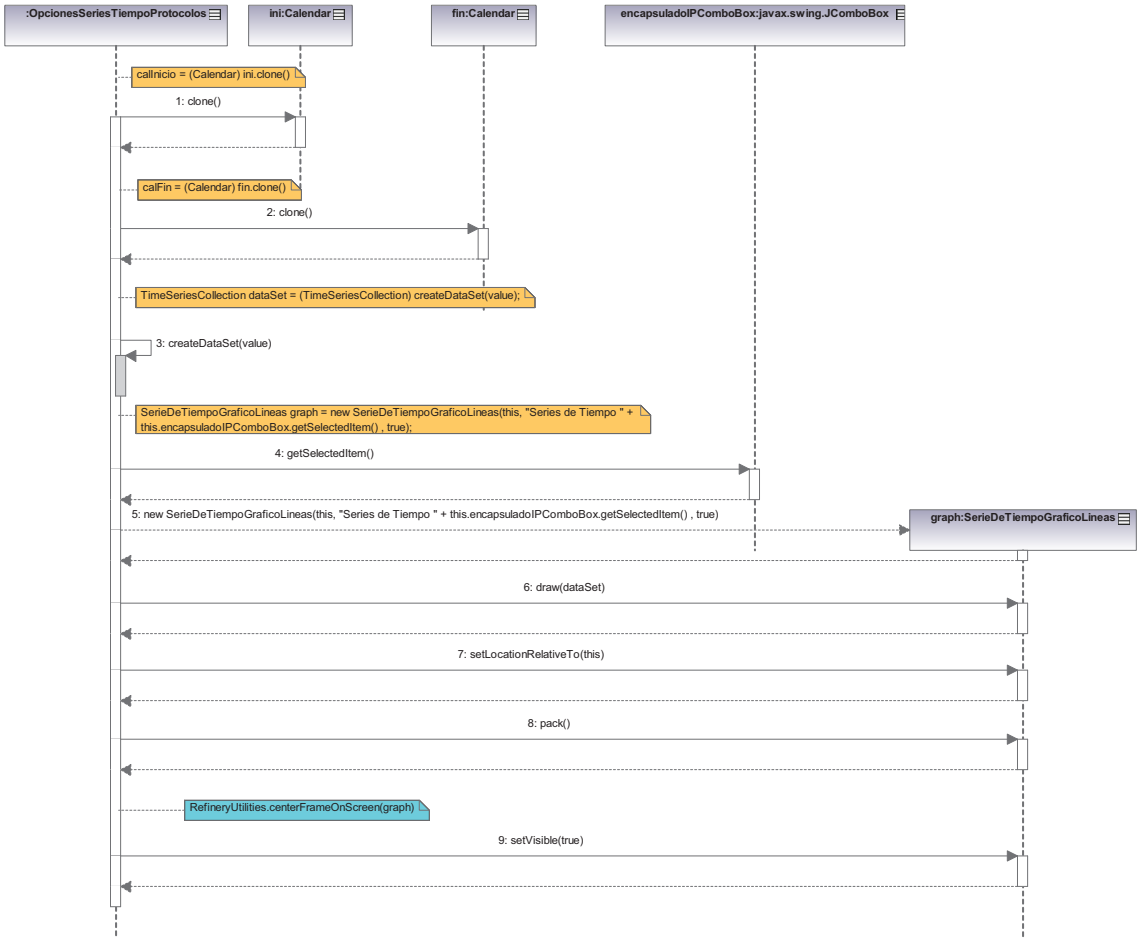


Gráfico 3.157 Diagrama de secuencia para obtener una gráfica de series de tiempo

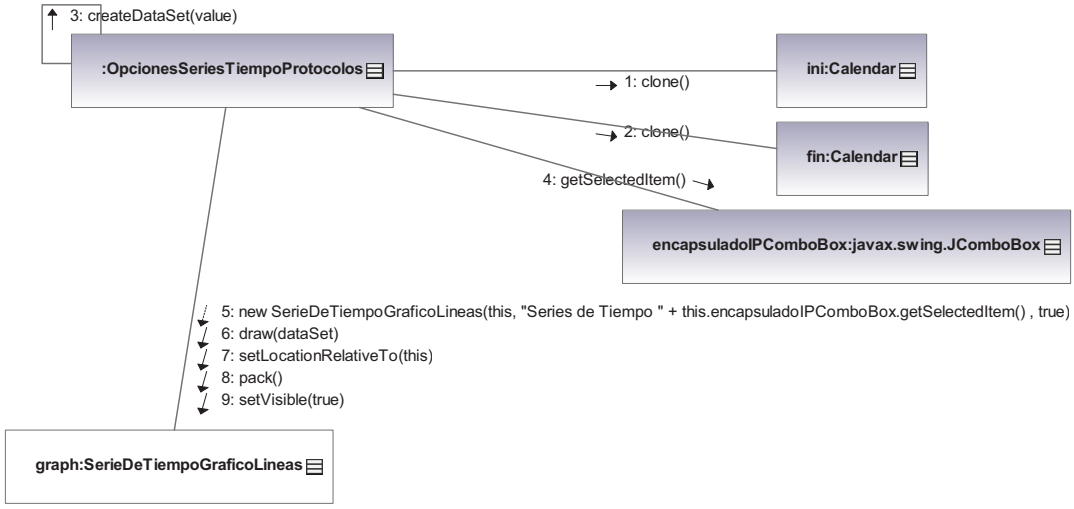


Gráfico 3.158 Diagrama de colaboración para obtener una gráfica de series de tiempo

OpcionesSeriesTiempoProtocolos	
Mensaje	Significado
1: clone() 2: clone()	Retorna una copia de un objeto Calendar con los mismos atributos y valores pero asignado en un diferente espacio de memoria.

3: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de las series de tiempo líneas.
4: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario
5: new SerieDeTiempoGraficoLineas(this, "Series de Tiempo " this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de series de tiempo líneas para los distintos protocolos y estaciones de trabajo.
6: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
7: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
8: pack()	Organiza y empaqueta los componentes gráficos para su representación.
9: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.28 Mensajes del diagrama de colaboración del gráfico 3.158

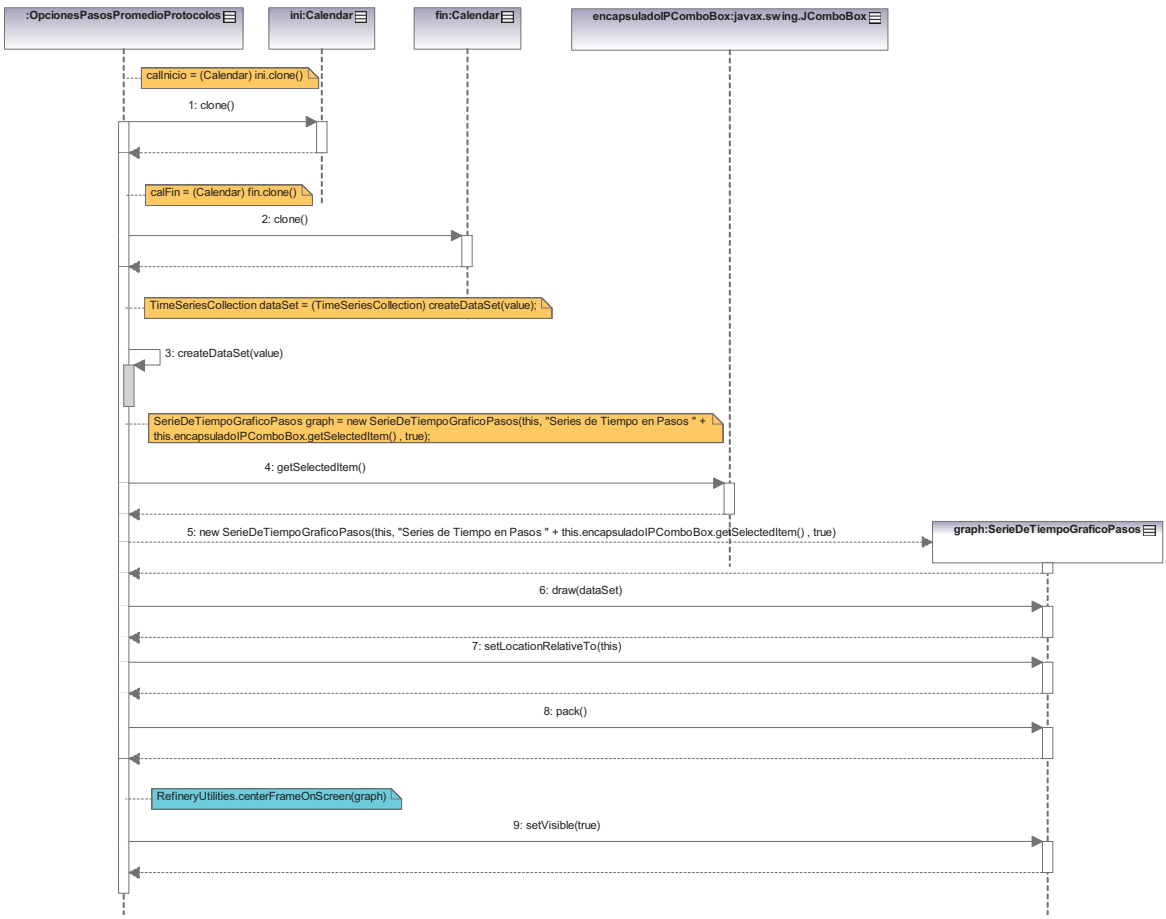


Gráfico 3.159 Diagrama de secuencia para obtener una gráfica de series de tiempo que usa pasos para cada intervalo de tiempo regular

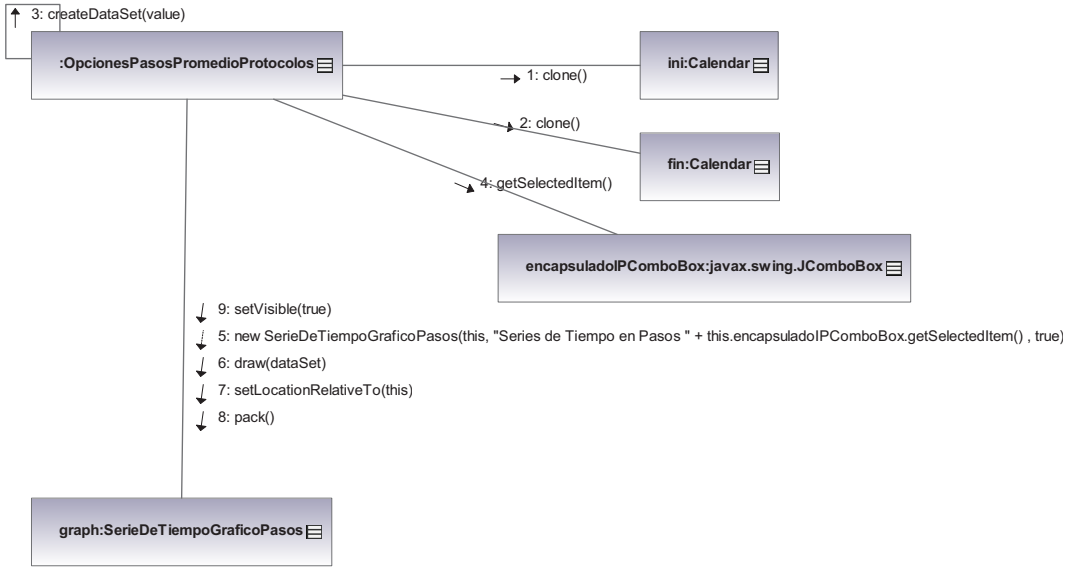


Gráfico 3.160 Diagrama de colaboración para obtener una gráfica de series de tiempo que usa pasos para cada intervalo de tiempo regular

OpcionesPasosPromedioProtocolos	
Mensaje	Significado
1: clone() 2: clone()	Retorna una copia de un objeto Calendar con los mismos atributos y valores pero asignado en un diferente espacio de memoria.
3: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de las series de tiempo en pasos.
4: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario
5: new SerieDeTiempoGraficoPasos(this, "Series de Tiempo en Pasos " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación de series de tiempo en pasos para los distintos protocolos y estaciones de trabajo.
6: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
7: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
8: pack()	Organiza y empaqueta los componentes gráficos para su representación.
9: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.29 Mensajes del diagrama de colaboración del gráfico 3.160

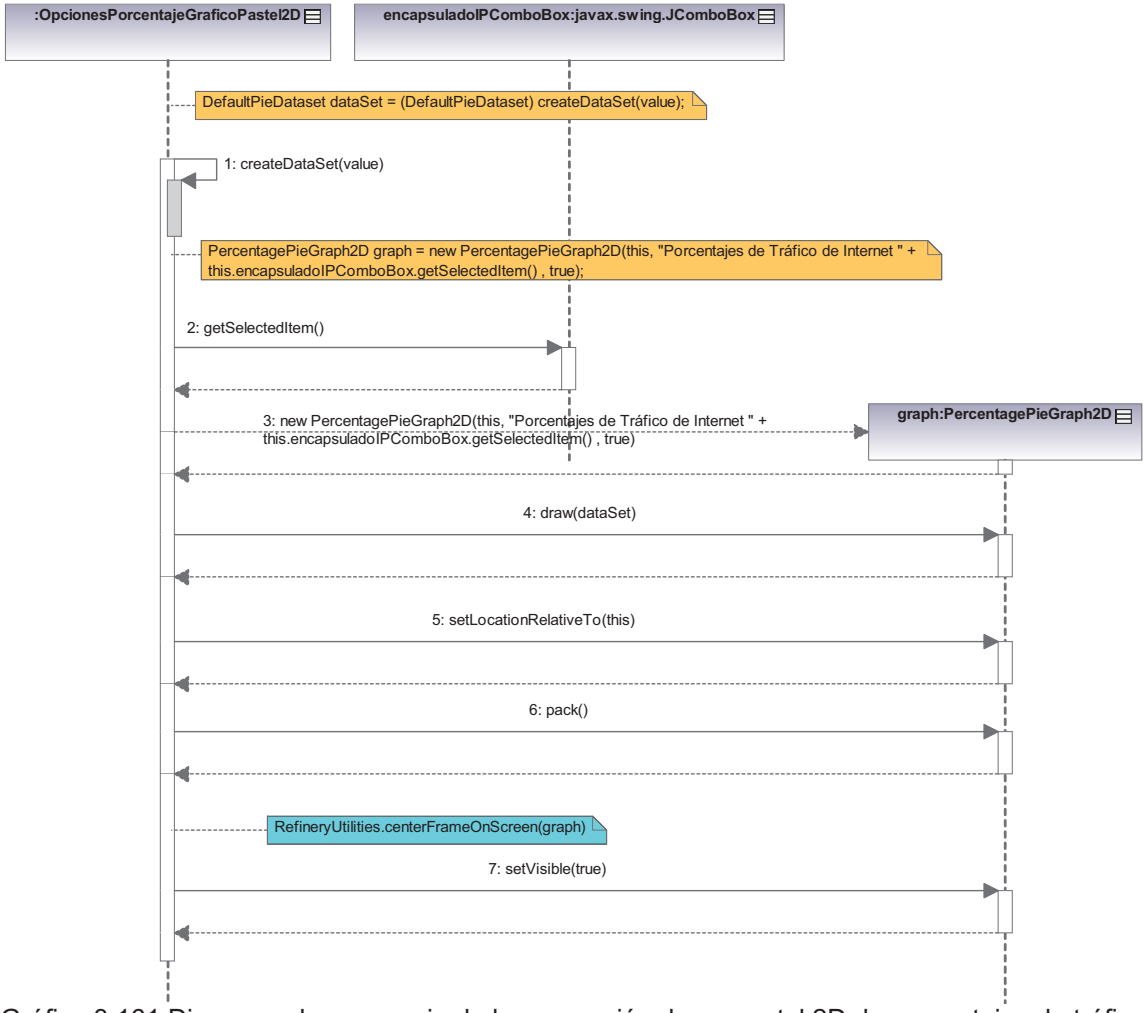


Gráfico 3.161 Diagrama de secuencia de la generación de un pastel 2D de porcentajes de tráfico de Internet

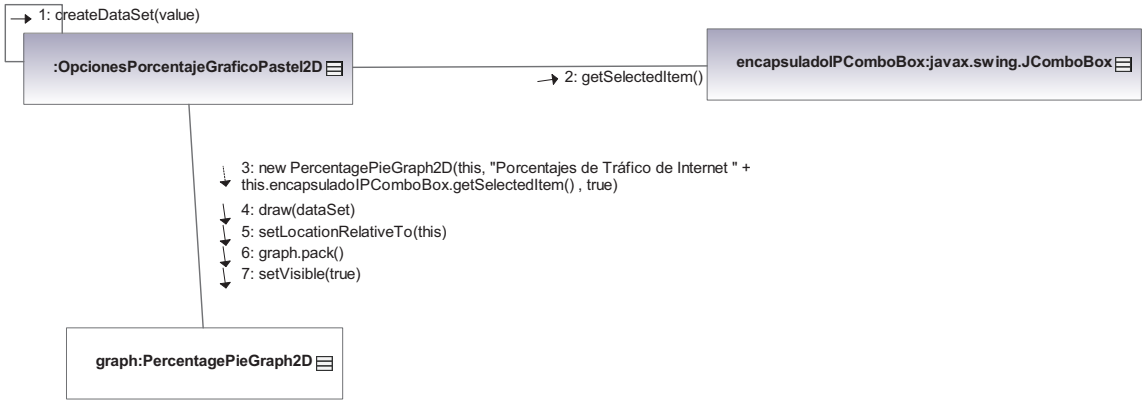


Gráfico 3.162 Diagrama de secuencia de la generación de un pastel 2D de porcentajes de tráfico de Internet

OpcionesPorcentajeGraficoPastel2D	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Tasa Promedio de Transferencia.
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.
3: new PercentagePieGraph2D(this, "Porcentajes de Tráfico de Internet " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación Porcentajes de Tráfico de Internet 2D para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: pack()	Organiza y empaqueta los componentes gráficos para su representación.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.30 Mensajes del diagrama de colaboración del gráfico 3.162

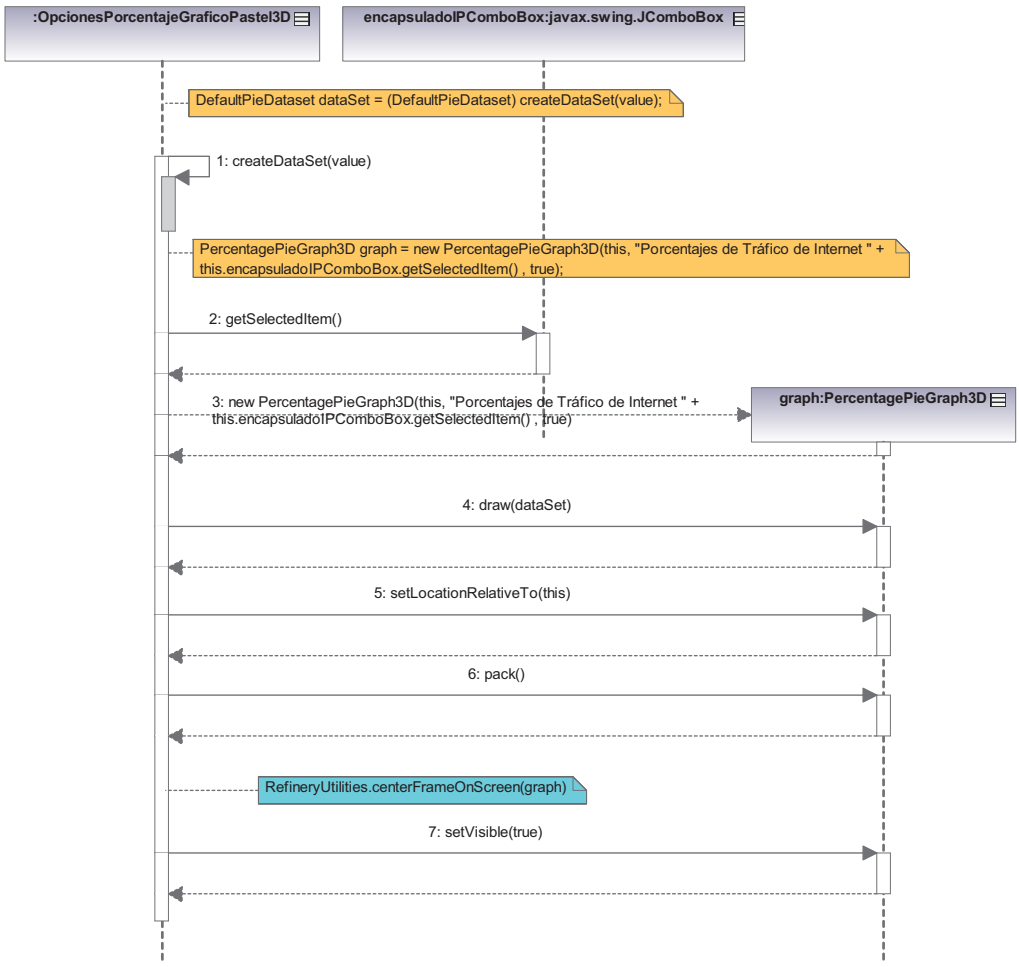


Gráfico 3.163 Diagrama de secuencia de la generación de un pastel 3D de porcentajes de tráfico de Internet

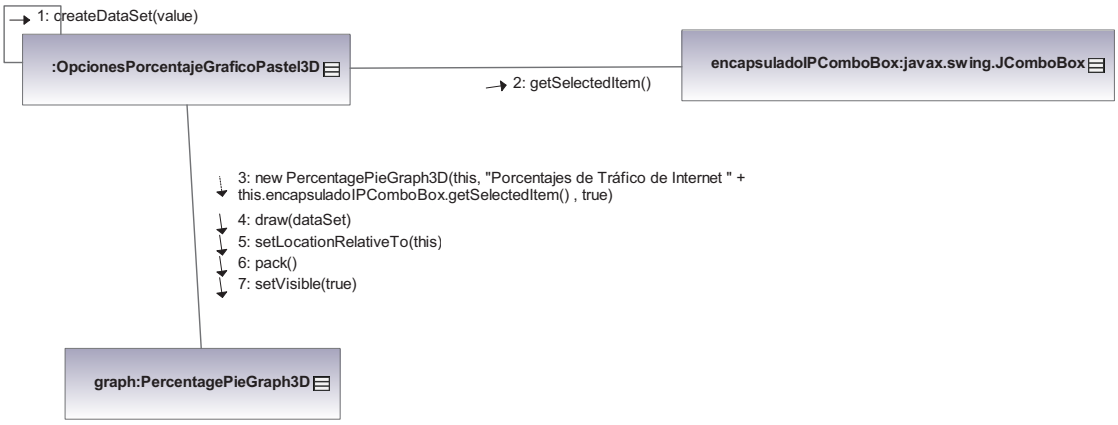


Gráfico 3.164 Diagrama de colaboración de la generación de un pastel 3D de porcentajes de tráfico de Internet

OpcionesPorcentajeGraficoPastel3D	
Mensaje	Significado
1: createDataSet(value)	Instancia una colección de datos para realizar la representación gráfica de Tasa Promedio de Transferencia.
2: getSelectedItem()	Retorna el tipo de protocolo para realizar la consulta a la base de datos de los hosts seleccionados por el usuario.
3: new PercentagePieGraph3D(this, "Porcentajes de Tráfico de Internet " + this.encapsuladoIPComboBox.getSelectedItem() , true)	Instancia un objeto que permite realizar la graficación Porcentajes de Tráfico de Internet 3D para los distintos protocolos y estaciones de trabajo.
4: draw(dataSet)	Instancia un panel principal que contendrá al gráfico obtenido de los datos de la colección
5: setLocationRelativeTo(this)	Permite graficar el cuadro de diálogo o la representación gráfica en el centro del objeto padre graficado.
6: pack()	Organiza y empaqueta los componentes gráficos para su representación.
7: setVisible(true)	Permite a los diferentes objetos gráficos ser visibles para el usuario.

Tabla 3.31 Mensajes del diagrama de colaboración del gráfico 3.164

Caso de uso:

- **Diferenciar tráfico (host, protocolos, puertos)**

La diferenciación de tráfico está dada por las distintas opciones que el usuario puede seleccionar en la ventana de diálogo que aparece una vez que se ha elegido el gráfico que se desea obtener. Una vez terminado el proceso de selección de opciones y se hace clic en aceptar se genera un evento que es capturado por la aplicación y tratado como se observa en el siguiente diagrama del gráfico 3.165.

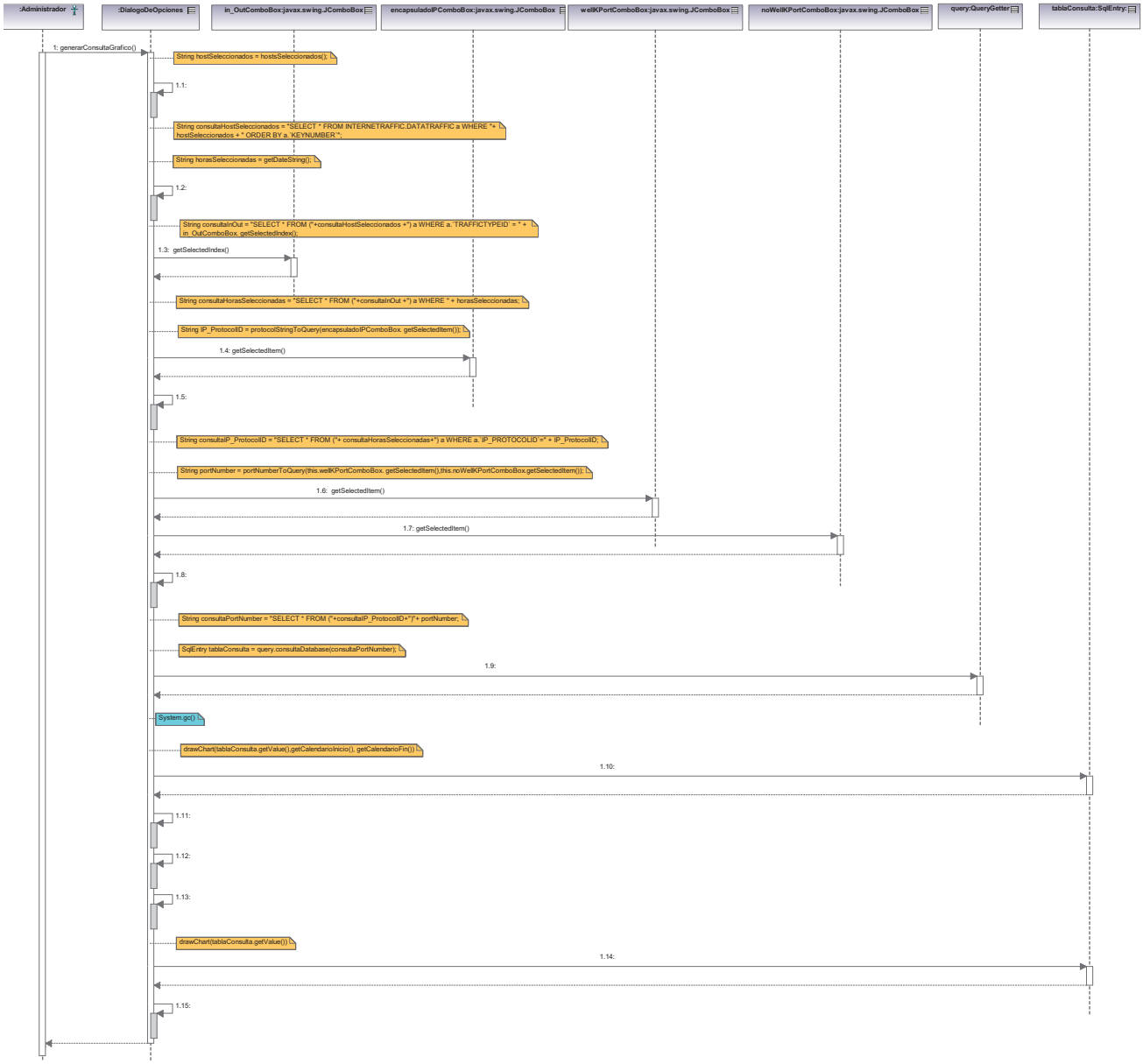


Gráfico 3.165 Diagrama de secuencia que muestra la diferenciación de tráfico según las opciones seleccionadas por el usuario, en la generación de una consulta SQL para la base de datos (método generarConsultaGrafico() de todas la clases que heredan de DialogoDeOpciones)

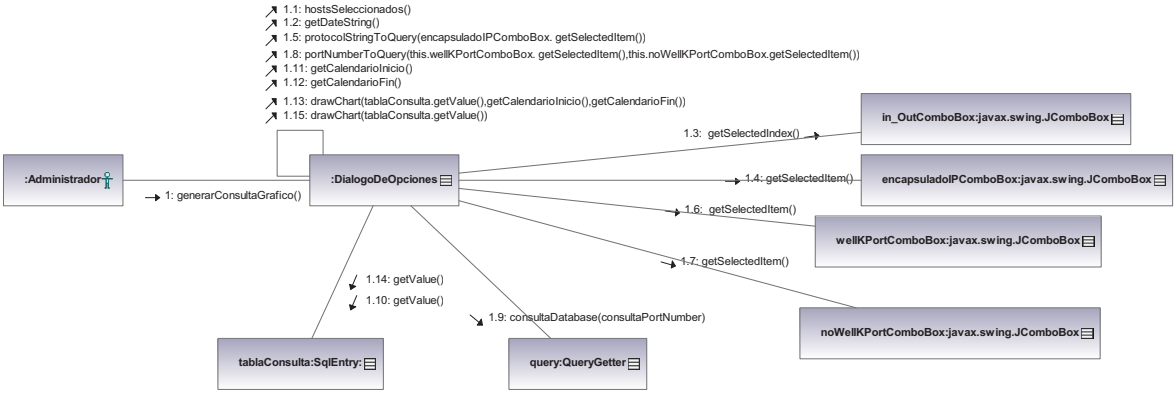


Gráfico 3.166 Diagrama de colaboración que muestra la diferenciación de tráfico según las opciones seleccionadas por el usuario, en la generación de una consulta SQL para la base de datos (método generarConsultaGrafico() de todas la clases que heredan de DialogoDeOpciones)

generarConsultaGrafico()	
Mensaje	Significado
1: generarConsultaGrafico()	Genera una cadena de caracteres con hosts, protocolos, encapsulado IP, tipo de tráfico, hora de inicio y fin en forma de consulta necesario para obtener los datos requeridos de la base de datos para cada diagrama estadístico.
1.1: hostsSeleccionados()	Retorna un objeto del tipo String que representa una cadena de caracteres de todos los hosts seleccionados lista para añadir a la sentencia de consulta para la base de datos.
1.2: getDateString()	Retorna un objeto del tipo String con el día inicio y fin en milisegundos para armar la consulta a la base de datos.
1.3: getSelectedIndex()	Retorna el índice del valor seleccionado del comboBox “Seleccione el tipo de tráfico” para diferenciar si es incoming o outgoing traffic.
1.4: getSelectedItem()	Retorna el índice del valor seleccionado del comboBox para diferenciar el tipo de protocolo ya sea UDP, TCP, ICMP.
1.5: protocolStringToQuery(encapsuladoIPComboBox. getSelectedItem())	Transforma el item seleccionado del ComboBox “Protocolo encapsulado IP” a una cadena de caracteres para la consulta a la base de datos.
1.6: getSelectedItem()	Retorna el índice del valor seleccionado del comboBox “Ptos Well Known TCP/UDP” para diferenciar el tipo de puertos bien conocidos.
1.7: getSelectedItem()	Retorna el índice del valor seleccionado del comboBox “Ptos No Well Known TCP/UDP” para diferenciar el tipo de puertos no bien conocidos.
1.8: portNumberToQuery(	Arma una cadena de caracteres para la

this.wellKPortComboBox. getSelectedItem(), this.noWellKPortComboBox. getSelectedItem())	consulta a la base de datos de acuerdo a la selección de puertos que el usuario haya elegido en el cuadro de diálogo.
1.9: consultaDatabase(consultaPortNumber)	Consulta a la base de datos para obtener un objeto del tipo Statement y otro objeto del tipo ResultSet relacionados unívocamente en un tercer objeto del tipo SqlEntry.
1.10: getValue() 1.14: getValue()	Retorna el ResultSet asociado.
1.11: getCalendarioInicio()	Retorna un objeto del tipo Calendar que representa al calendario inicio del cuadro de diálogo.
1.12: getCalendarioFin()	Retorna un objeto del tipo Calendar que representa al calendario fin del cuadro de diálogo.
1.13: drawChart(tablaConsulta.getValue(),getCalendarioInicio(),getCalendarioFin())	Métodos a implementar según requiera el diagrama elegido por el usuario (abstract). Se encargará de organizar los datos de la consulta para representarlos gráficamente, especialmente para los diagramas temporales.
1.15: drawChart(tablaConsulta.getValue())	Métodos a implementar según requiera el diagrama elegido por el usuario (abstract). Se encargará de organizar los datos de la consulta para representarlos gráficamente.

Tabla 3.32 Mensajes del diagrama de colaboración del gráfico 3.166

Caso de uso:

- **Almacenar y recuperar de disco**

  El proceso de adquisición de datos desde la base es usado casi de manera implícita durante la generación de los gráficos, como paso posterior e inmediato a la obtención de la cadena de texto de consulta. Este importante proceso para el cumplimiento del presente caso de uso es realizado usando los métodos del paquete database.

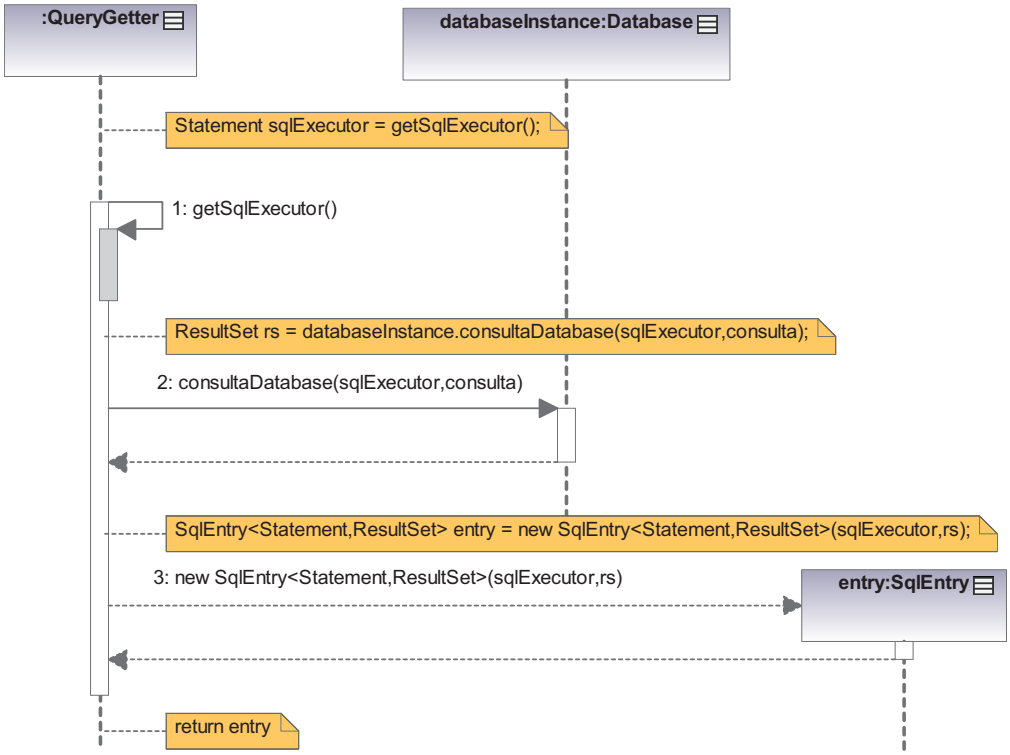


Gráfico 3.167 Diagrama de secuencia para la realización de una consulta previo establecimiento de la conexión con la base en QueryStatistics por medio del método consultaDatabase(String consulta) de QueryGetter.

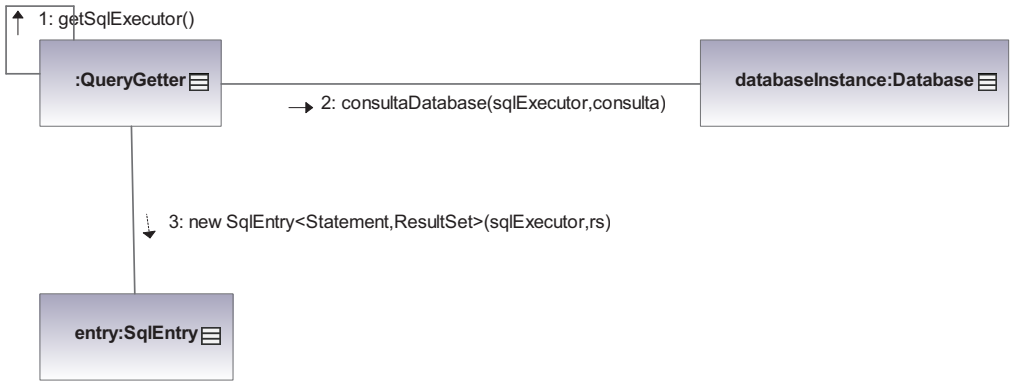


Gráfico 3.168 Diagrama de secuencia para la realización de una consulta previo establecimiento de la conexión con la base en QueryStatistics por medio del método consultaDatabase(String consulta) de QueryGetter.

consultaDatabase(in consulta:String)	
Mensaje	Significado
1: getSqlExecutor()	Retorna un objeto del tipo Statement de la base de datos actual.
2: consultaDatabase(sqlExecutor,consulta)	Consulta a la base de datos a través de un objeto Statement.
3: new SqlEntry<Statement,ResultSet>(sqlExecutor,rs)	Instancia un objeto SqlEntry que mantiene una relación unívoca entre un objeto Statement y un objeto ResultSet.

Tabla 3.33 Mensajes del diagrama de colaboración del gráfico 3.168

CAPÍTULO 4

4. IMPLEMENTACIÓN DEL PROTOTIPO, PRUEBAS DE MONITOREO Y ANÁLISIS DE COSTOS

Concluido el proceso de diseño y desarrollo se generó una versión distribuible del proyecto. Posteriormente se describirán las respectivas pruebas de funcionamiento para verificar el cumplimiento de los objetivos y corregir posibles errores. También se presentan varios escenarios para demostrar la utilidad del proyecto. De acuerdo al plan propuesto también se mostrará comparaciones con respecto a otro software similar junto con un análisis de costos de desarrollo del proyecto.

4.1. DESCRIPCIÓN DE LOS PAQUETES EJECUTABLES Y LOS PAQUETES DE BIBLIOTECAS EN LA DISTRIBUCIÓN FINAL DEL SOFTWARE DEL PROYECTO DE TITULACIÓN

El paquete distribuible consta de los siguientes elementos:

Windows:

```
TrafficAndQueryStatistics
   +
   +-->TrafficStatistics.jar: Aplicación para la captura de paquetes y sniffer.
   +
   +-->QueryStatistics.jar: Aplicación para generar gráficos estadísticos.
   +
   +-->jnetpcap.dll: Intérprete entre jNetPcap y Windows.
   +
   +--lib: Bibliotecas.
      +--> TrafficStatisticsLibrary.jar: Biblioteca desarrollada en el proyecto.
      +
      +--> appframework-1.0.3.jar: Swing Application Framework NetBeans.
      +
      +--> DJNativeSwing-SWT.jar: Provee componentes para NativeSwing.
      +
      +--> DJNativeSwing.jar: Integración de componentes nativos y
      +                       aplicaciones Swing.
      +--> jna-3.0.7.jar: Acceso de Java a librerías compartidas de Windows.
      +
      +--> jna_WindowUtils.jar: Encapsula varias utilidades para Windows.
      +
```

+--> jnetpcap-1.2.rc5.jar: Biblioteca Java intérprete WinPcap y libpcap.
+--> jsc.jar: Biblioteca estadística.
+
+--> MozillaInterfaces-1.8.1.3.jar: Biblioteca para interactuar con el
+ navegador MozillaFirefox.
+--> mysql-connector-java-5.1.7-bin.jar: Administra base de datos
+ MySQL desde Java.
+--> mysql-connector-mxj-gpl-5-0-9-db-files.jar: Binarios de la base de
+ datos MySQL para diferentes sistemas operativos.
+
+--> mysql-connector-mxj-gpl-5-0-9-fixed.jar: Embebe una base de
+ datos MySQL en una aplicación Java.
+
+--> substance.jar: Plantillas visuales para aplicaciones Java.
+
+--> swing-worker-1.1.jar: Manejo de eventos.
+
+--> swt-3.6M3-win32-win32-x86.jar: Interfaz gráfica dependiente de la
+ plataforma.
+--> gnujaxp.jar: Implementación del estándar XML para APIs de Java.
+
+--> iText-2.1.5.jar: Biblioteca base para JFreeChart.
+
+--> JCalendar.jar: Calendario visual para aplicaciones Java.
+
+--> jcommon-1.0.16.jar: Biblioteca base para JFreeChart.
+
+--> jfreechart-1.0.13-experimental.jar: Biblioteca base para
+ JFreeChart.
+--> jfreechart-1.0.13-swt.jar: Biblioteca base para JFreeChart.
+
+--> jfreechart-1.0.13.jar: Biblioteca gratuita para generación de
+ gráficos en aplicaciones Java.
+--> junit.jar: Unidad de prueba para framework.
+
+--> swtgraphics2d.jar: Extensión de Graphics2D usado por
 JFreeChart.

Para la distribución de CentOS 5.2 y Ubuntu 8.04 se incluyen los paquetes jnetpcap-1.2.rc5-fc8.i386.rpm y jnetpcap-1.2.rc5-deb.i386.deb respectivamente que contiene a jNetPcap.

Una biblioteca adicional común para Linux es swt-3.6-gtk-linux-x86.jar para el uso de elementos nativos de sistemas operativos basados en Unix.

4.2. COMPARACIÓN DEL SOFTWARE DESARROLLADO CON WIRESHARK WIN32-1.2.8 Y COLASOFT CAPSA 7.1

La necesidad de los administradores de red para alcanzar el máximo rendimiento con los menores recursos posibles, ha llevado a los desarrolladores de soluciones de software orientados a la red a implementar herramientas que permitan suplir esta necesidad, para administrar eficientemente los recursos de red.

En la actualidad el Internet se ha convertido en un valioso recurso y para su monitoreo, existen un sin número de aplicaciones de las cuales se elegirán dos para realizar la respectiva comparación con el proyecto de titulación. Siguiendo el plan propuesto anteriormente las aplicaciones elegidas son Wireshark win32-1.2.8 y Colasoft Capsa 7.1 Full Demo 30 Days.

Se comparará todas las características afines entre los dos programas y el proyecto de titulación de manera general por medio de tablas.

4.2.1. Comparación de requerimientos mínimos de hardware y software.

La tabla 4.1 muestra los requerimientos mínimos de las tres aplicaciones, detallando el sistema operativo, hardware y software adicional que no venga en sus instaladores.

Items	**Proyecto de Titulación**	**Wireshark win32-1.2.8**	**Colasoft Capsa 7.1**
Procesador	P4 1.6 GHz/ AMD Atlon 1.2 GHz	P3 700 MHz	P4 2.8 GHz
Memoria RAM	1 GB	128 MB	2 GB
Browser	Mozilla o Internet Explorer 6.0	Mozilla o Internet Explorer 6.0	Internet Explorer 6
Aplicaciones Adicionales	Abode Flash Player 10 ActiveX	Ninguna	Ninguna
Sistema Operativo	Windows XP SP2 Windows 2003 server Windows Vista Windows 7 CentOS 5.2 Ubuntu 8.02	Windows XP SP2 Windows 2003 server Windows Vista Windows 7 CentOS 5.2 Ubuntu 8.02	Windows XP SP2 Windows 2003 server Windows Vista Windows 7

Tabla 4.1 Requerimientos Mínimos del Sistema

4.2.2. Comparación de características afines del software desarrollado con Wireshark win32-1.2.8 y Colasoft Capsa 7.1

La tabla 4.2 se muestra una breve comparación de las características afines entre las tres aplicaciones. Para profundizar más en las características de comparación se sugiere leer el anexo D.

Características	Proyecto de Titulación	Capsa Colasoft 7.1	Wireshark win32-1.2.8
CAPTURA DE DATOS			
Selección Interfaz de red	SI	SI	SI
Lista de estaciones de trabajo monitoreadas	SI	SI	NO
Browser	SI	NO	NO
Filtro por Protocolo	SI	SI	SI
Filtro por Puerto	SI	SI	SI
Descripción de campos de cabeceras de paquetes	SI	SI	SI
Gráficos en Tiempo real del tráfico total	SI	SI	SI(Limitado)
Gráficos en Tiempo real del tráfico por hosts seleccionados	SI(Elección Simple)	SI(Configurando Subredes)	SI (Implementando filtros)
Gradiente de color para graficación del bitrate en tiempo real.	SI	NO	NO
Tray icons para tráfico total en tiempo real	SI	NO	NO
Tray icons para tráfico por hosts seleccionados.	SI	SI	SI
Diferenciación de tráfico entrante y saliente en graficación de tiempo real	SI	NO	NO
Reinicio de gráficos en tiempo real para trafico entrante y saliente	SI	NO	NO
Captura de gráficos en tiempo real	SI	SI	NO
Alarmas	SI	SI	NO
Asignación de valores críticos de bitrate	SI	NO	NO
Guardar contenido de paquetes decodificados	SI	SI	SI
Importar y exportar base de datos	SI	NO	NO
ANÁLISIS DE DATOS			
Tasa de transferencia promedio de uso de Internet	SI	SI	NO
Porcentaje de uso de tráfico de Internet 2D	SI	SI	NO
Porcentaje de uso de	SI	NO	NO

tráfico de Internet 3D			
Histograma de frecuencias de bitrates calculados en un rango de tiempo	SI	NO	NO
Distribución de frecuencias acumuladas de protocolos de tráfico de Internet.	SI	NO	NO
Reconstrucción de historial de la base de datos usando líneas	SI	NO	NO
Reconstrucción de historial de la base de datos usando pasos	SI	NO	NO
Series de tiempo	SI	NO	NO
Serie de tiempo en pasos	SI	NO	NO
Ranking para IPs más utilizadas	SI	SI	NO
DNS Reverso	SI	SI	NO
Reproductor de Video Para tutorial	SI	NO	NO
APARIENCIA (GUI)			
Look & feel (Plantillas de mejoramiento visual)	SI	NO	NO
Animaciones Flash para descripción de herramientas de análisis de datos.	SI	NO	NO

Tabla 4.2 Comparación del Proyecto de titulación con respecto a dos aplicaciones afines

4.3. PRUEBAS DE MONITOREO

Con la finalidad de comprobar formalmente si el proyecto cumple con los objetivos planteados al inicio de este trabajo, se realizaron pruebas de monitoreo en el Laboratorio de Software perteneciente a la Facultad Ingeniería en Sistemas que da servicios a la ESFOT.

Este laboratorio cuenta con 20 máquinas para prestar servicio de Internet, y 4 máquinas adicionales para administrar la red, las direcciones IP son asignadas por el servidor DHCP de la Escuela Politécnica Nacional.

Para poder monitorear el tráfico de Internet se utilizaron los siguientes equipos.

- Laptop Sony Vaio con características de 2.2 GHz Dual Dore y 3 GB de RAM, con una tarjeta de red Ethernet.

- Un switch de 8 puertos D-Link no administrable.
- Un hub de 8 puertos de 10 Mbps.
- 5 computadores de la Intranet.

La conexión de red requerida para el monitoreo de datos se la realizó conectando 5 computadoras de la intranet a un switch no administrable D-link, este a su vez se conecta a un puerto del hub.

Para que estas estaciones de trabajo puedan salir a Internet se conecta un puerto del hub a un puerto del switch administrable de la Intranet.

La computadora portátil se conecta directamente al hub para monitorear todo el tráfico que genera las 5 estaciones de trabajo. El gráfico 4.1 permite describir la conexión necesaria para el monitoreo de los datos.

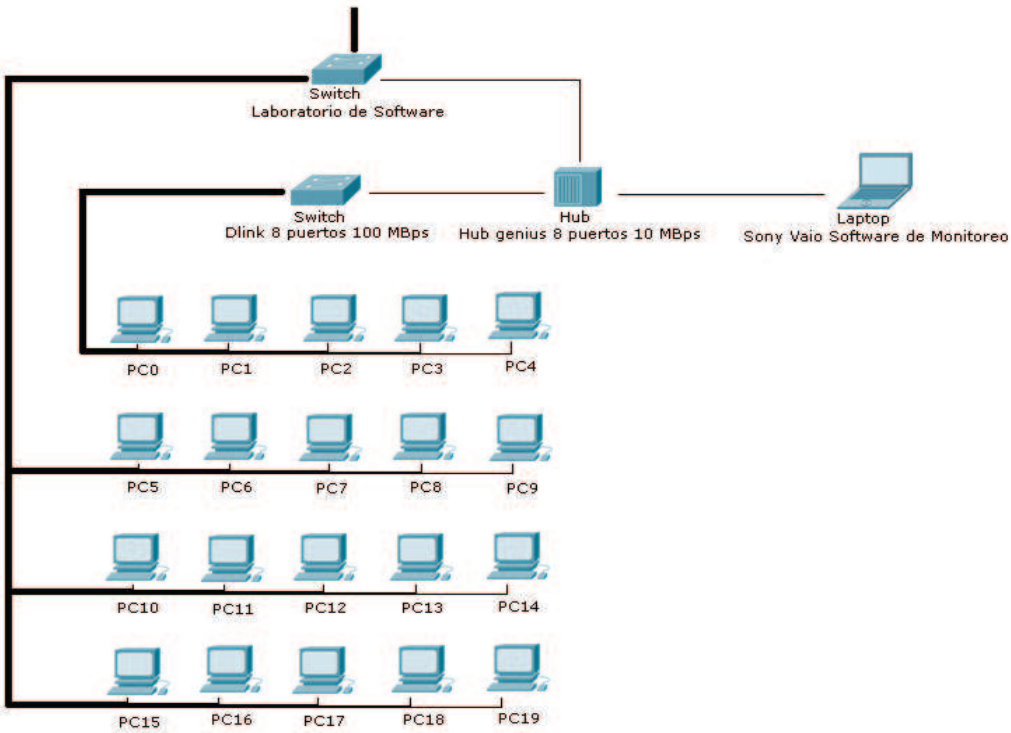


Gráfico 4.1 Diagrama de conexión de red para monitoreo de datos.

Luego de haber implementado el esquema de conexión física anterior, se procedió a monitorear el tráfico de Internet los días designados por el responsable del laboratorio de software que fueron el día jueves 27 y viernes 28 de mayo del 2010.

4.3.1. Captura de paquetes y almacenamiento en la base de datos

En los gráficos 4.2 y 4.3 se muestra la captura de datos obtenida de un puerto del hub en tiempo real, mostrando un árbol en la parte izquierda del gráfico con todas las estaciones de trabajo monitoreadas. Ambas capturas de pantalla muestran el gráfico de bitrate vs tiempo con una alarma a 100 KBps, mostrando en detalle la tasa de transferencia instantánea y promedio del tráfico entrante y saliente.

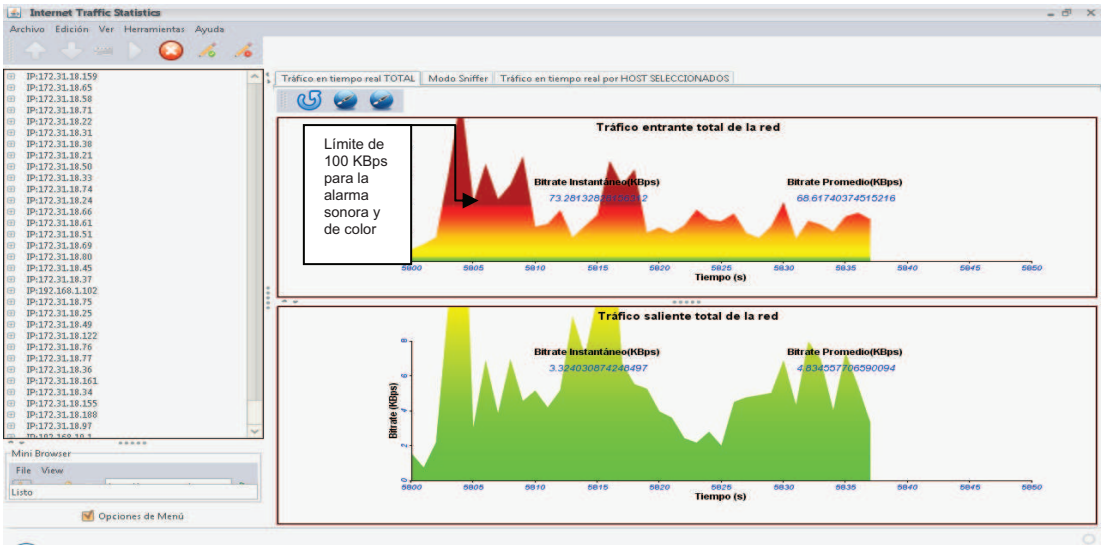


Gráfico 4.2 Gráfico bitrate vs tiempo incoming y outgoing de las pruebas de monitoreo sobrepasando los 100KBps.

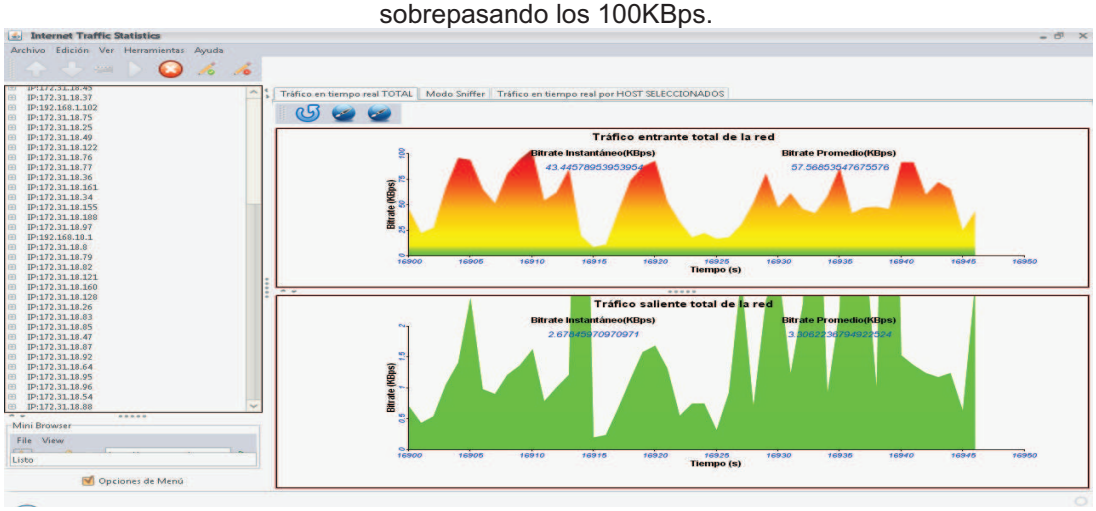


Gráfico 4.3 Gráfico bitrate vs tiempo incoming y outgoing de las pruebas de monitoreo en un momentos diferente.

Cuando las estaciones de trabajo dejaban de transmitir y recibir datos el gráfico bitrate vs tiempo deja de representar valores como se muestra en el gráfico 4.4.

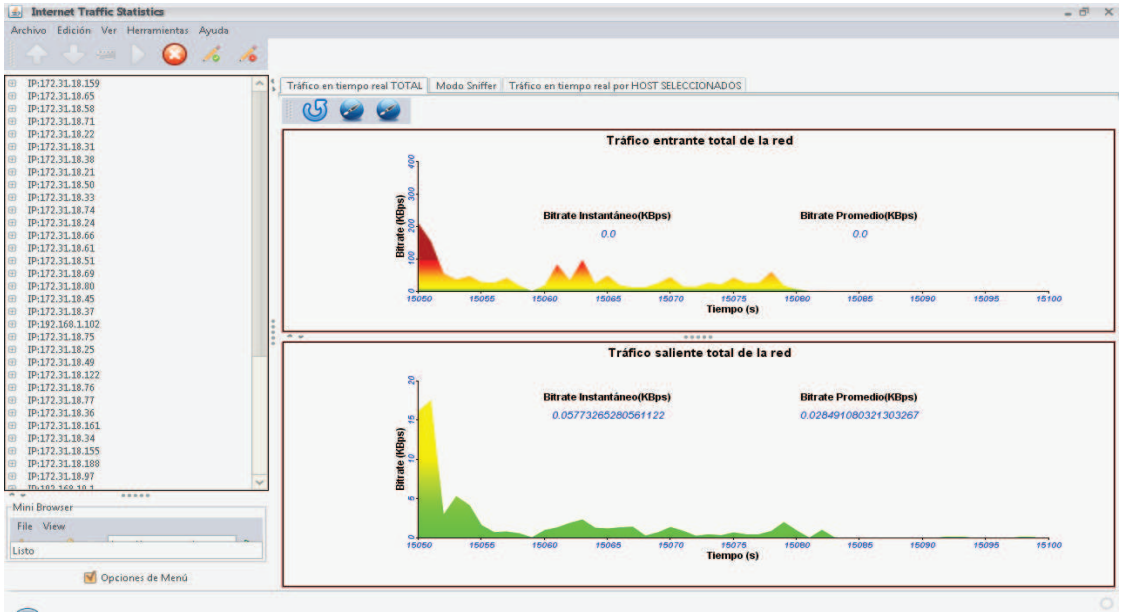


Gráfico 4.4 Gráfico bitrate vs tiempo incoming y outgoing cuando termina una descarga de datos.

Para analizar el tráfico de un host específico se procedió a señalar la casilla de activación sobre una estación de trabajo y como se muestra en la siguiente captura de pantalla del gráfico 4.5, el programa muestra solo el tráfico que generaba la estación seleccionada al utilizar la conexión a Internet.

Gráfico 4.5 Gráfico bitrate vs tiempo incoming y outgoing para un host seleccionado.

Para visualizar el flujo del tráfico de varios hosts, se señaló en varias casillas de activación en el árbol de estaciones de trabajo situado en la parte izquierda. Luego de inicializar el sniffer el programa empezó a graficar los valores de bitrate

como se muestra en la siguiente captura de pantalla representada por el gráfico 4.6.

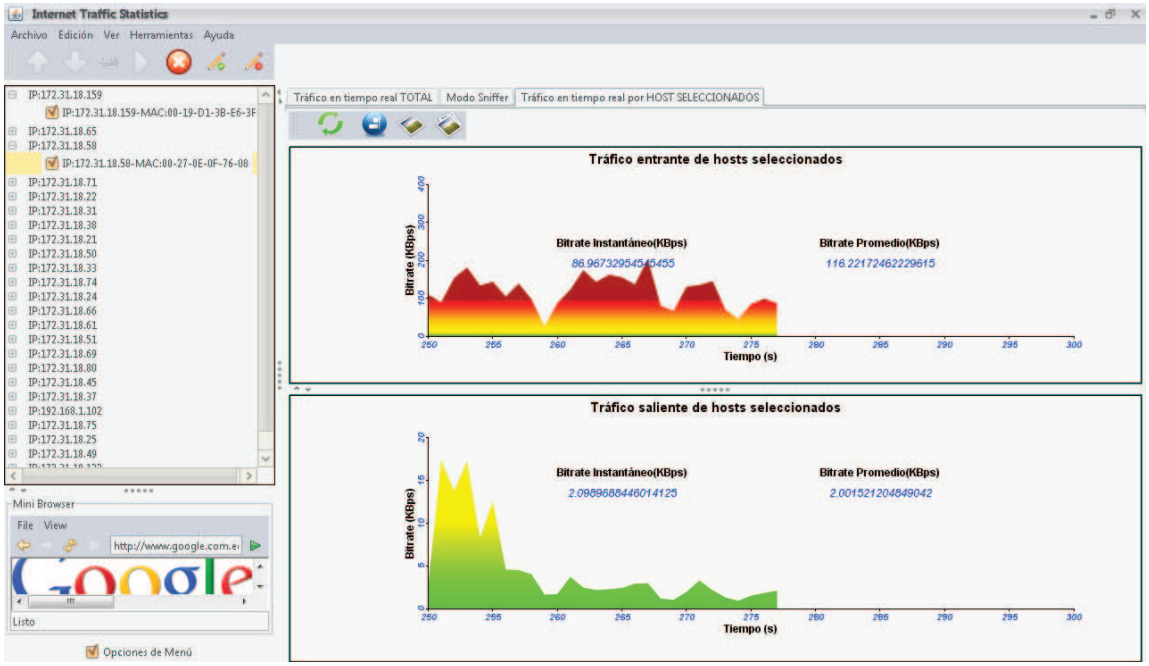


Gráfico 4.6 Gráfico bitrate vs tiempo incoming y outgoing para varios hosts seleccionados.

Cuando se inicializa en análisis de datos por hosts seleccionados el sniffer empieza a capturar paquetes y a representarlos en un área de texto, como se muestra en la siguiente captura de pantalla del gráfico 4.7.

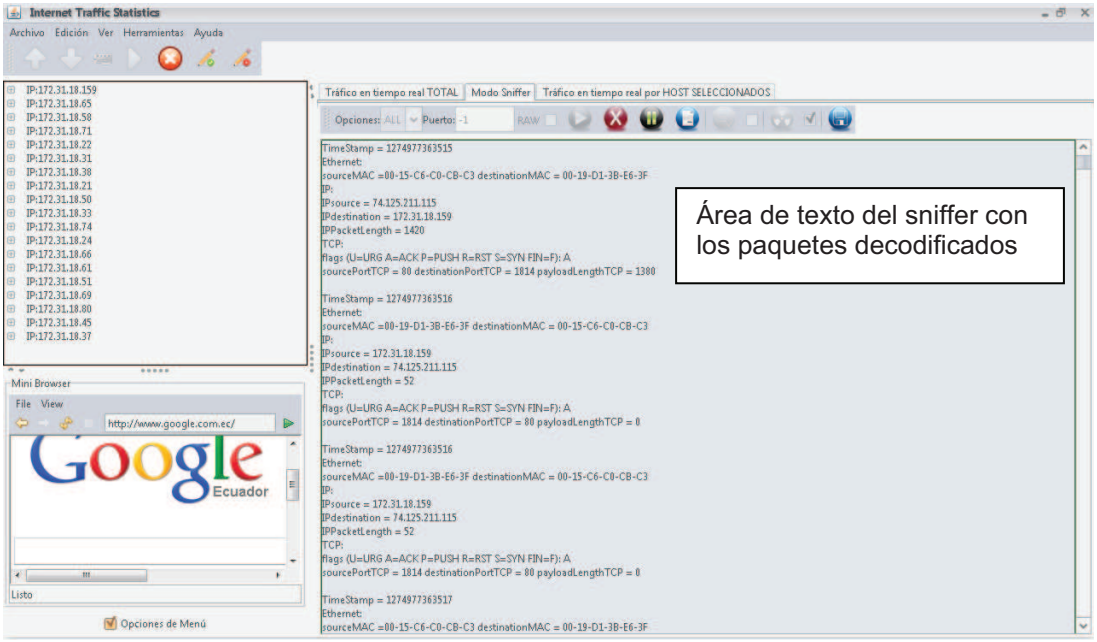


Gráfico 4.7 Monitoreo de paquetes en Modo Sniffer.

Si al inicializar el sniffer no se ha asignado ningún tipo de opción para almacenar o para descartar los datos monitoreados despliega en pantalla una opción para guardar los datos monitoreados en un archivo de texto, como se muestra en la siguiente captura de pantalla representada por el gráfico 4.8.

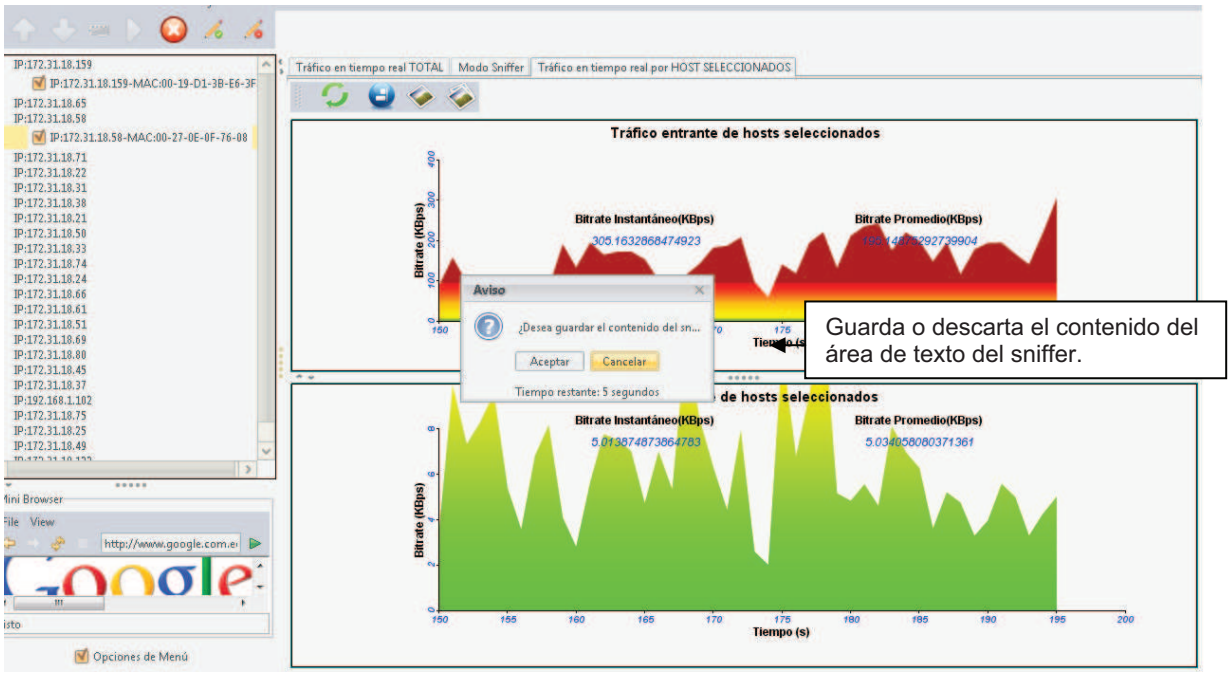


Gráfico 4.8 Cuadro de diálogo para respaldar datos monitoreados en Modo Sniffer.

Al presionar el botón “aceptar” del cuadro de diálogo anterior, aparece una ventana para elegir un directorio y el nombre del archivo de texto que contendrá los paquetes decodificados del tráfico de Internet.

Como se ha visto casi en todas las capturas se ha añadido un browser al proyecto, para que el usuario pueda navegar en Internet y acceder a los servidores de la intranet.

A continuación se muestran los paquetes decodificados del archivo de texto que generó el software al guardar la información de captura de tráfico de Internet. Deshabilitando la casilla en la barra de herramientas de la pestaña “Sniffer” se consigue asignar el modo NO RAW (decodificación de cabeceras de protocolo) y al activar la misma casilla en la barra de herramientas se asigna el modo RAW (decodificación completa del paquete) para la captura de paquetes.

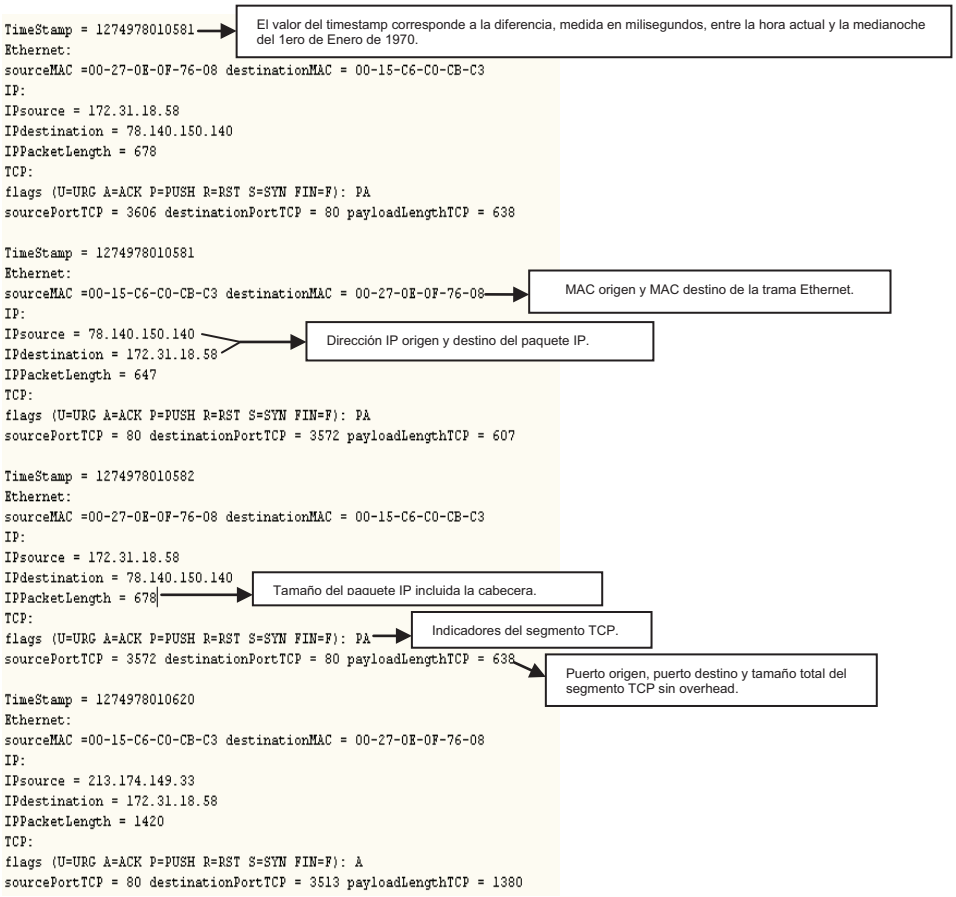


Gráfico 4.9 Valores de campos de paquetes en modo de decodificación No RAW.

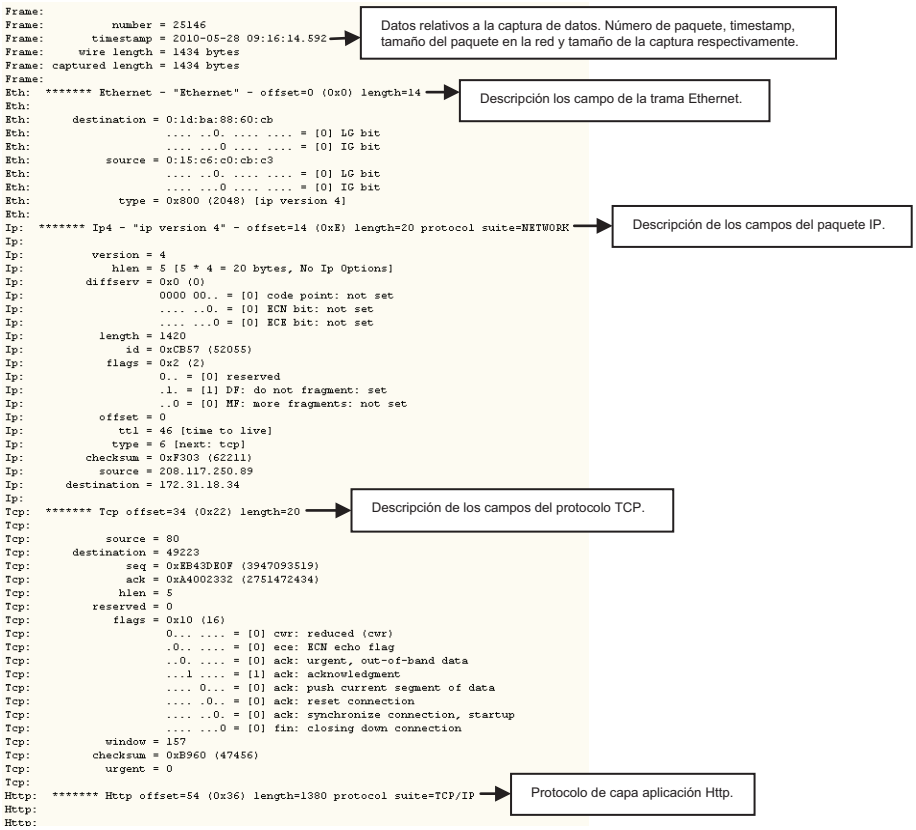


Gráfico 4.10 Valores de campos de paquetes en modo de decodificación RAW.

4.3.2. Análisis de resultados

Después de haber monitoreado durante 2 días la intranet del Laboratorio de Software se procedió a realizar el análisis respectivo.

Para profundizar en la descripción de cada tipo de gráfico y resumen de datos de tráfico de Internet expuesto a continuación, se recomienda leer el anexo D.

4.3.2.1. Tasa de transferencia promedio de uso de Internet [1]

Con este diagrama se examinó los datos monitoreados y se eligieron los hosts con los datos transferidos más significativos, para así comparar los diferentes valores de tasa transferencia del tráfico entrante de datos desde Internet.

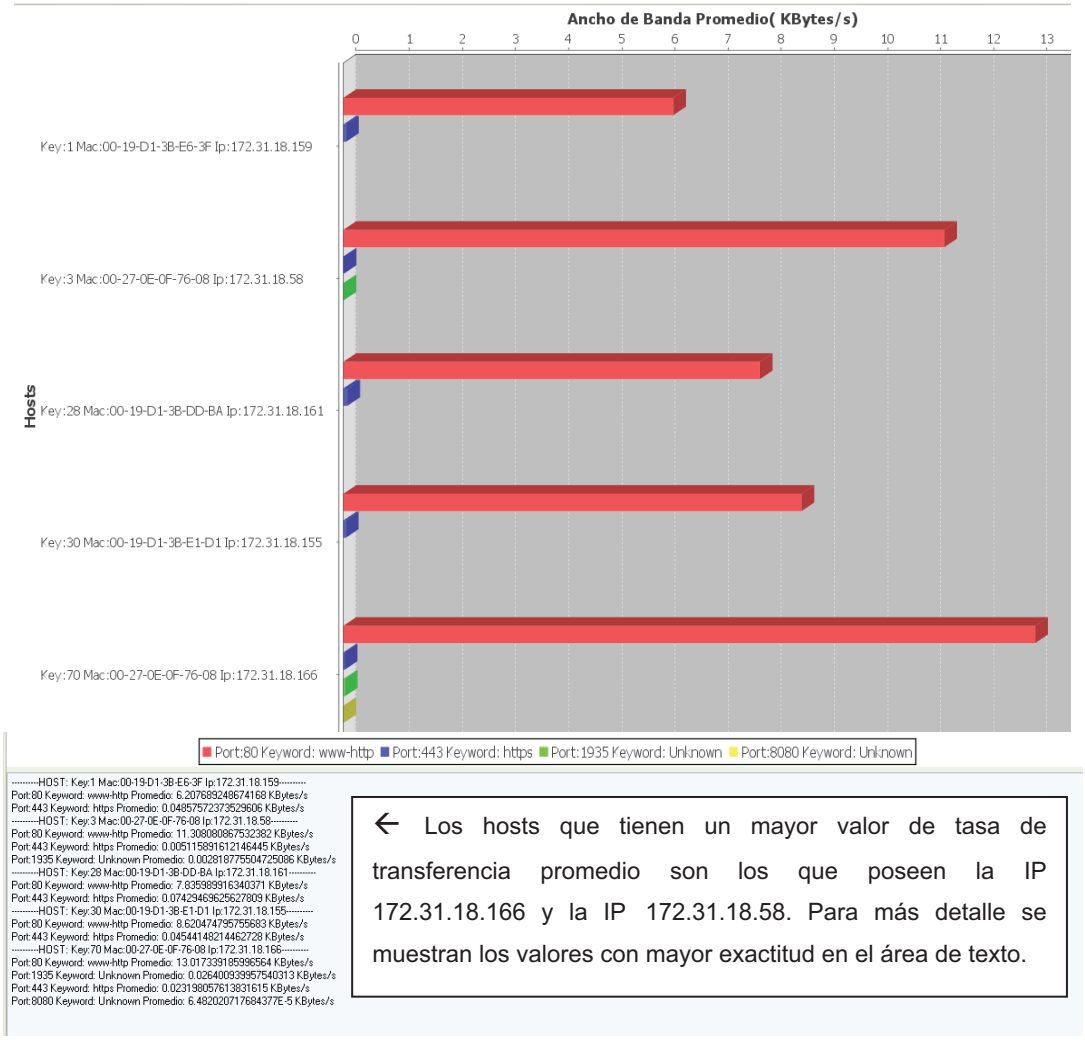


Gráfico 4.11 Tasa de transferencia promedio de uso de Internet de los datos monitoreados.

[1] Ver anexo D, sección D.1.7.1.1.

Como se muestra en el gráfico 4.11, se ve que ha existido una mayor cantidad de tráfico por el puerto 80, que corresponde al protocolo http, en menor magnitud para el puerto 443 que corresponde al protocolo https y para los puertos no bien conocidos 8080 y 1935.

4.3.2.2. Porcentajes de uso de tráfico de Internet [2]

Este diagrama permite representar la cantidad total de datos entrantes o salientes para los protocolos TCP, UDP, ICMP o para un puerto específico en un pastel de porcentajes.

Como se muestra en la representación gráfica 4.12, el host con la IP 172.31.18.166 posee el 28% del total con la mayor cantidad de transferencia de datos TCP del tráfico monitoreado, seguido por el host con la IP 172.31.18.58 con un porcentaje del 24%.

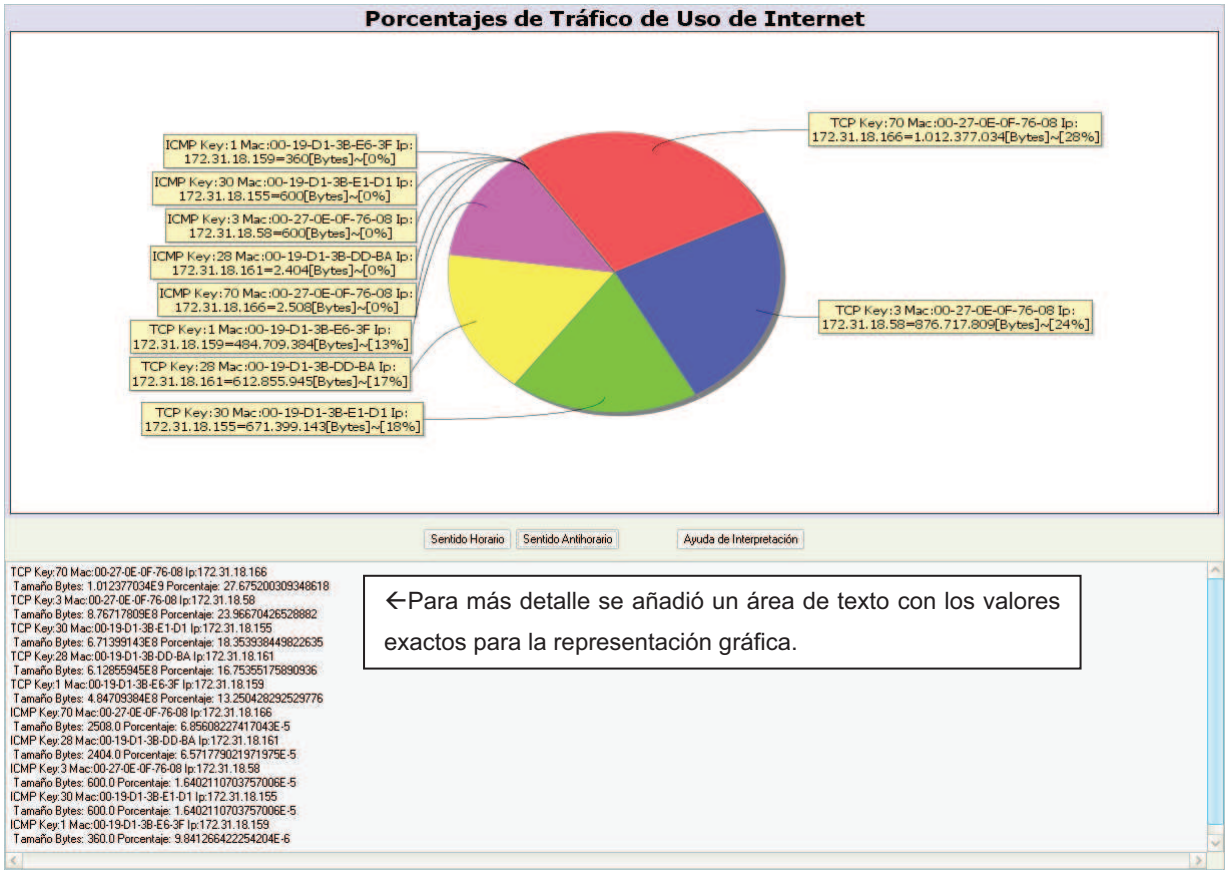


Gráfico 4.12 Porcentajes de uso de tráfico de Internet de los datos monitoreados.

[2] Ver anexo D, sección D.1.7.1.2

4.3.2.3. Porcentaje de uso de tráfico de Internet 3D [3]

Este diagrama servirá para mostrar con más detalle los valores del tráfico a Internet para los hosts seleccionados. En este caso se muestra la minoría del tráfico monitoreado en la siguiente captura representada por el gráfico 4.13.

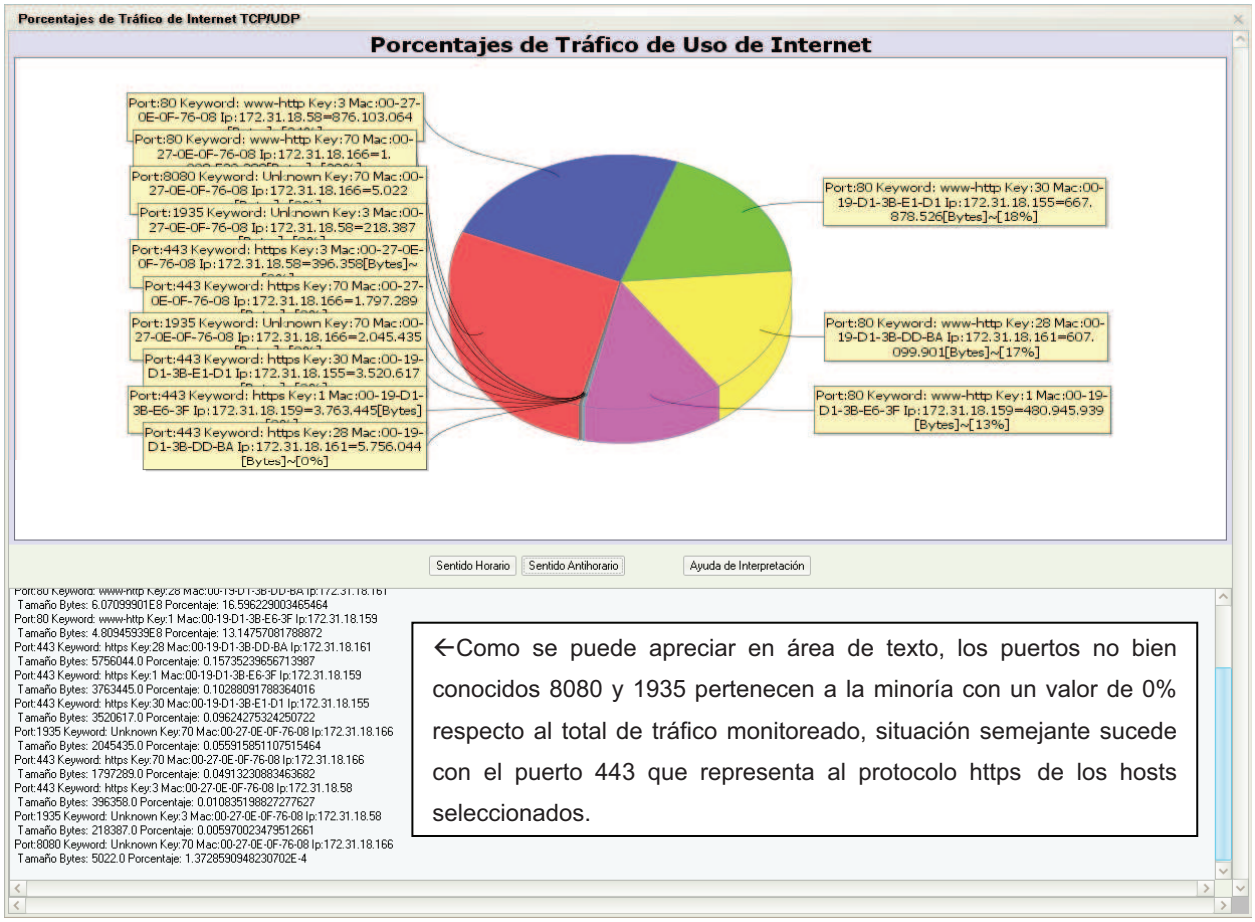


Gráfico 4.13 Porcentajes de uso de tráfico de Internet 3D de los datos monitoreados.

4.3.2.4. Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet [4]

Este diagrama realiza un análisis con los valores de tasa de transferencia para clasificarlos en rangos y obtener información sobre el comportamiento de la red.

---

[3] Ver anexo D, sección D.1.7.1.3
[4] Ver anexo D, sección D.1.7.1.4

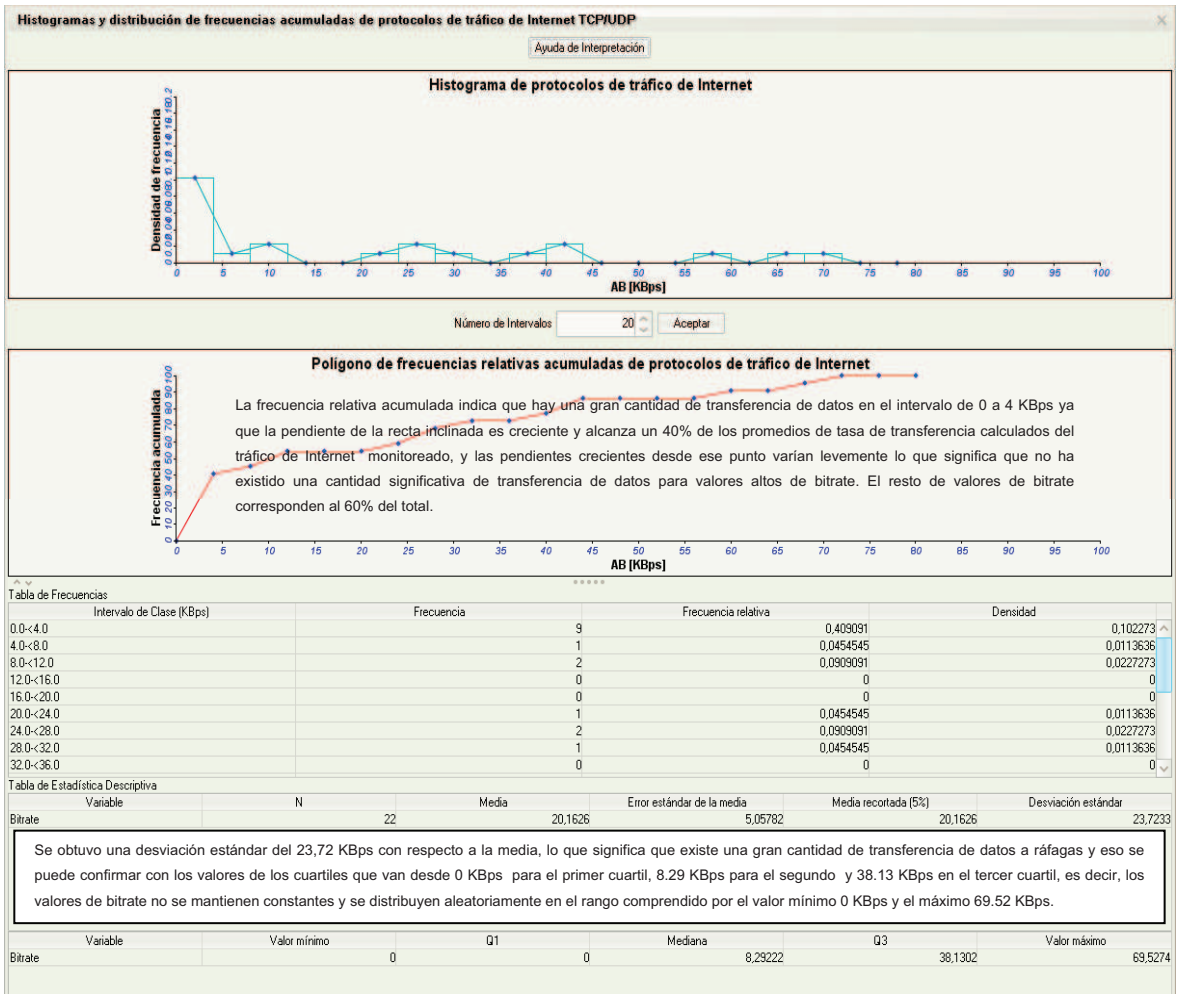


Gráfico 4.14 Histograma de protocolos y Distribución de frecuencias relativas acumuladas de los datos monitoreados.

En el gráfico 4.14 representa una herramienta que permite mostrar que ha existido una gran cantidad de transferencia de datos que oscilan entre 0 y 4 KBps, esto significa que los usuarios han estado navegando y descargando archivos pequeños.

Por otro lado las frecuencias a 10, 26 y 43 KBps muestran una descarga de archivos. Por la magnitud de su frecuencia se concluye que este comportamiento no fue constante.

La tabla de frecuencia indica los intervalos, la frecuencia relativa y la densidad de los datos monitoreados, los cuales permiten mostrar con más detalle los valores representados en el histograma y el polígono de frecuencias acumuladas.

La tabla de estadística descriptiva permite mostrar que las estaciones de trabajo durante los dos días de monitoreo han tenido un promedio de descarga del 20.16 KBps; rechazando los datos atípicos al 5% se obtiene el mismo valor de descarga del 20.16 KBps, esto significa que no han existido valores pico que distorsionen el valor del promedio.

4.3.2.5. Reconstrucción de historial de la base de datos con líneas.[5]

El diagrama permite representar los datos de bitrate cada 20 segundos que se encuentran almacenados en la base de datos.

Esto permite ayudar a identificar el tráfico que existió para las estaciones de trabajo seleccionadas.

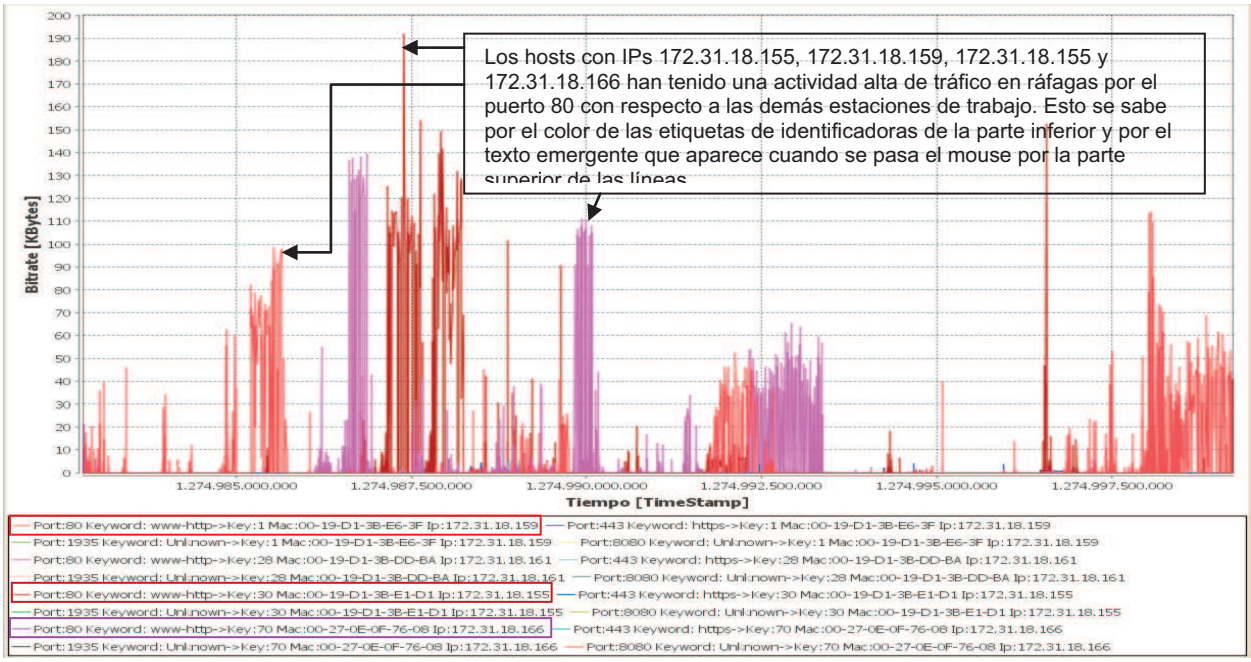


Gráfico 4.15 Reconstrucción de historial de la base con líneas de los datos monitoreados.

4.3.2.6. Reconstrucción de historial de la base de datos en pasos [6]

El diagrama es una gran herramienta para saber en qué horas del día un host o varios hosts han utilizado la conexión a Internet y en qué medida.

---

[5] Ver anexo D, sección D.1.7.1.5
[6] Ver anexo D, sección D.1.7.1.6

Además el gráfico muestra que de 11:00 a 17:30 del día jueves y desde las 9:30 hasta las 13:50 del día viernes horas ha existido una gran cantidad de transferencia de información desde Internet.

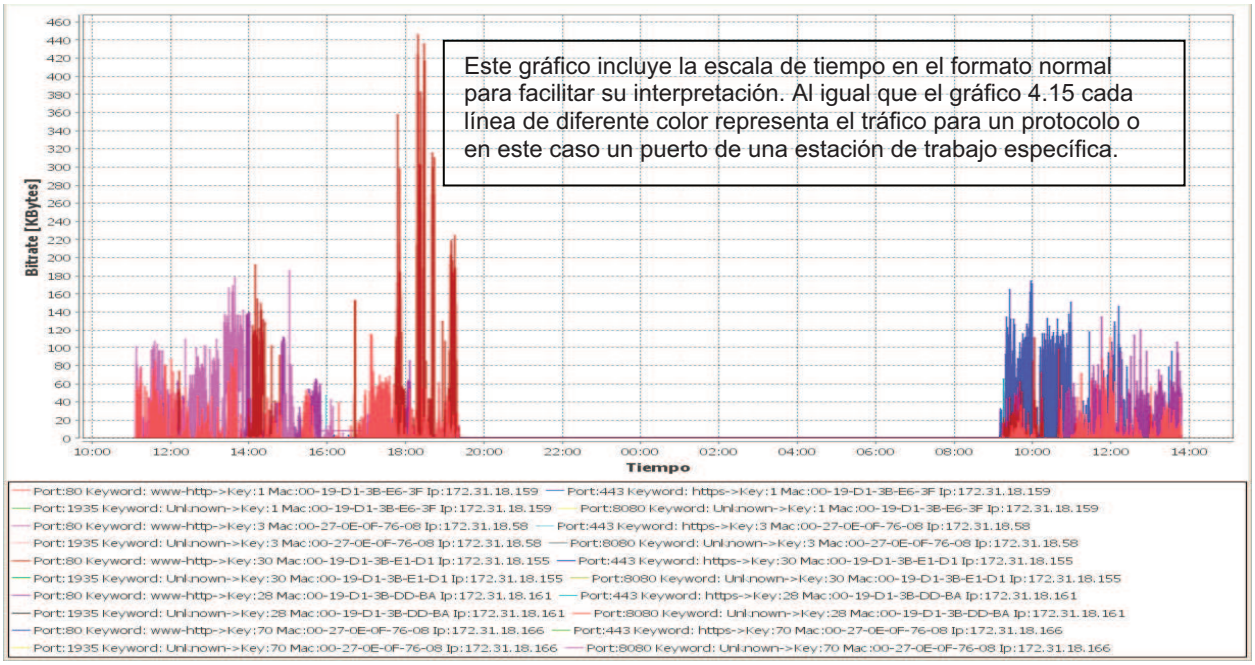


Gráfico 4.16 Reconstrucción de historial de la base en pasos de los datos monitoreados.

4.3.2.7. Series de tiempo [7]

El diagrama permite graficar el bitrate promedio calculado para intervalos de tiempo regulares.

El análisis de los datos obtenidos muestra que el host con dirección IP 172.31.18.159 y el host con la IP 172.31.18.161 han tenido una actividad alta por el puerto 80 con respecto a los demás hosts.

[7] Ver anexo D, sección D.1.7.1.7

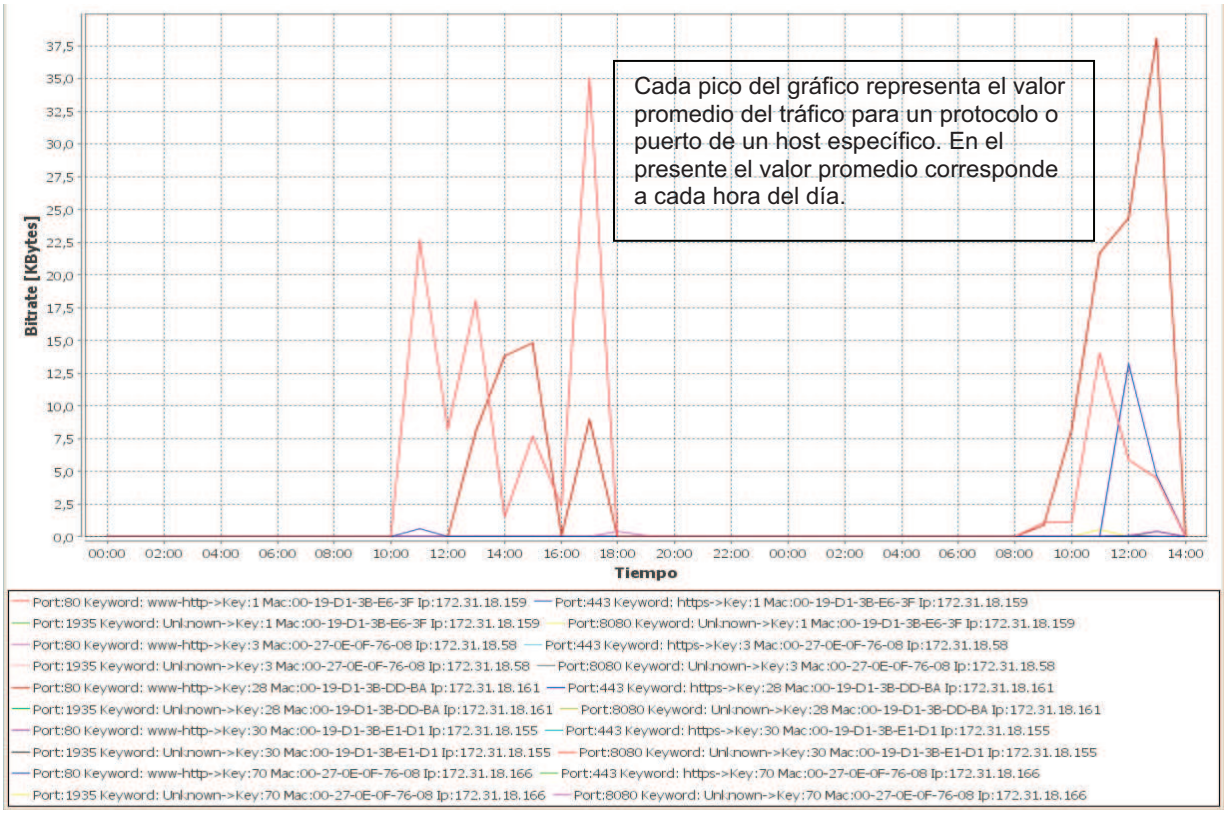


Gráfico 4.17 Series de tiempo de los datos monitoreados.

4.3.2.8. Series de tiempo en pasos [8]

El diagrama permite graficar el bitrate promedio calculado en pasos para intervalos de tiempo regulares.

El análisis de los datos muestra que el host con dirección IP 172.31.18.159 y el host con la IP 172.31.18.161 han tenido una actividad alta de transferencia de datos por el puerto 80 con respecto a los demás hosts.

[8] Ver anexo D, sección D.1.7.1.8

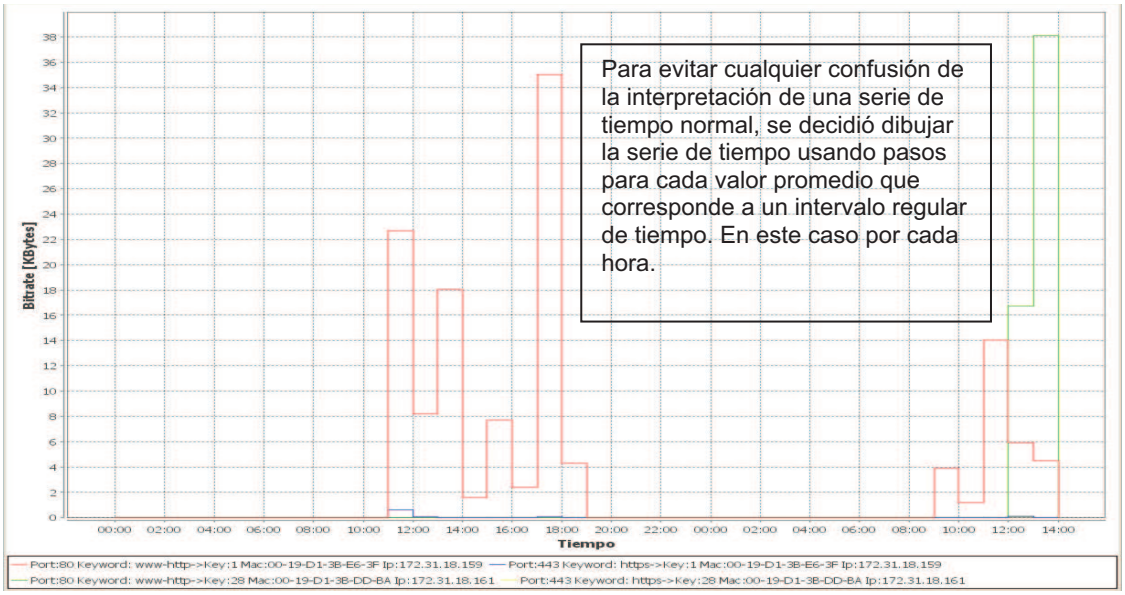


Gráfico 4.18 Series de tiempo en pasos de los datos monitoreados.

4.3.2.9. DNS reverso y ranking para IPs más utilizadas [9]

El diagrama permite obtener información ordenada de la cantidad de bytes transmitidos y recibidos por cada host que haya seleccionado el usuario, y de la misma manera realizar la resolución de nombres para las direcciones IPs más utilizadas.

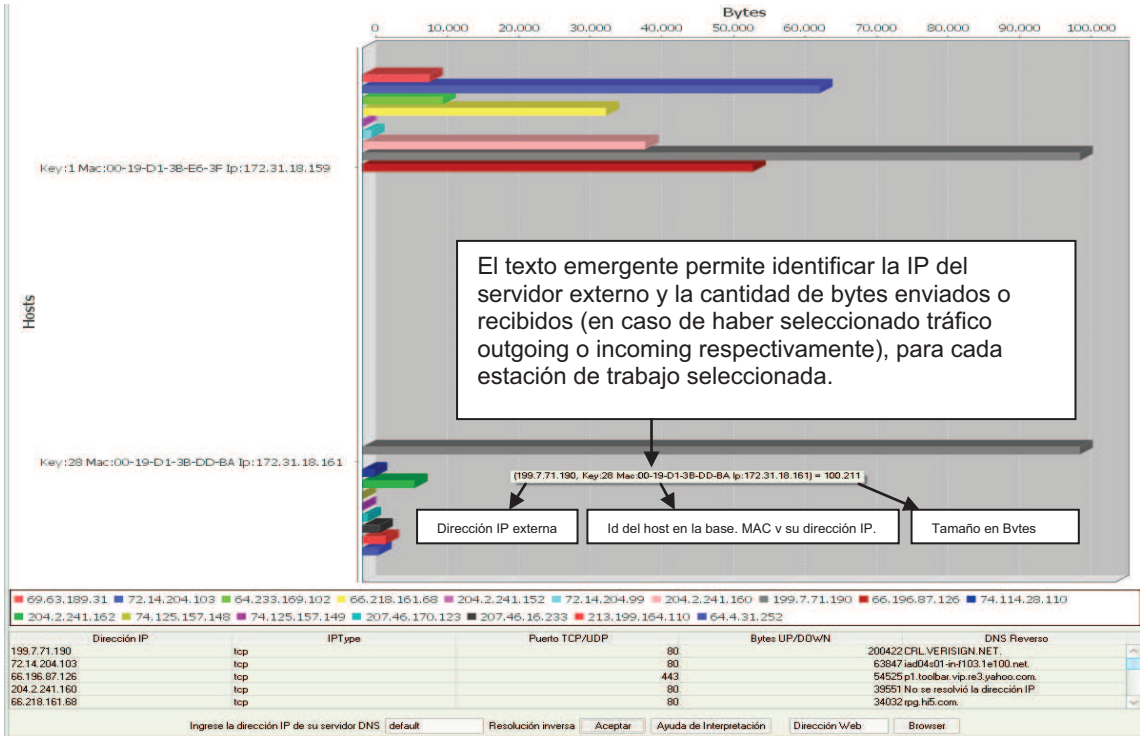


Gráfico 4.19 DNS reverso y ranking para IPs más utilizadas de los datos monitoreados.

[9] Ver anexo D, sección D.1.7.1.9

En el gráfico 4.19 se puede apreciar que los hosts con direcciones IP 172.31.18.159 y 172.31.18.161 han generado una considerable descarga de datos, estos a su vez se han conectado a varios servidores, siendo el de mayor transferencia de bytes el que posee la dirección IP 199.7.71.190 como se muestra en la parte superior de la tabla.

Además el software permite realizar la resolución inversa para todas las direcciones externas monitoreadas y así dar a conocer su nombre de dominio con las cuales tuvieron actividad de transferencia de datos. Para la dirección IP 199.7.71.190 su dominio es CRL.VERISIGN.NET.

4.4. ESCENARIOS DE DETECCIÓN DE ANOMALÍAS EN EL USO DEL SERVICIO DE INTERNET

En esta sección se presentarán varios escenarios los cuales permitirán mostrar la utilidad del proyecto de titulación como una herramienta para la detección de anomalías en el uso del servicio de Internet.

4.4.1. Primer Escenario – Detección de descargas no autorizadas.

Con los datos obtenidos del monitoreo de tráfico de Internet se requiere saber si existe algún comportamiento inusual como pueden ser descargas directas de una o de varias estaciones de trabajo, las cuales ocasionan que el acceso a Internet sea lento.

Para este tipo de análisis se requiere utilizar la herramienta “Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet “.

Se asigna la fecha y las horas en donde existió utilización de acceso a Internet para todas las estaciones de trabajo.

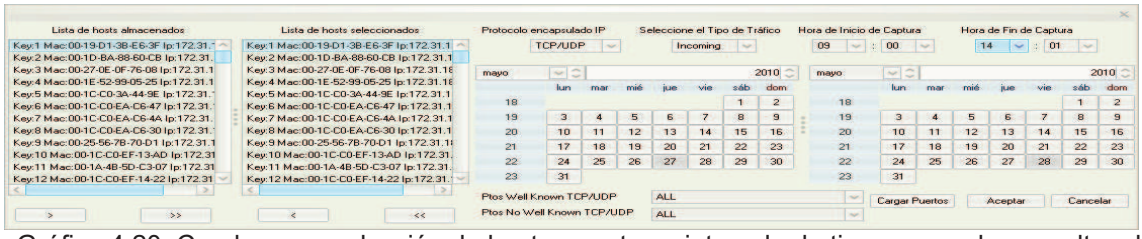


Gráfico 4.20 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos (Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet).

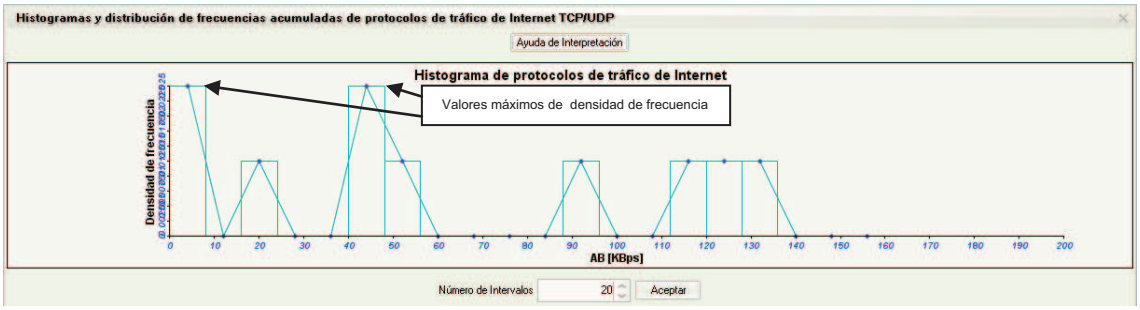


Gráfico 4.21 Histograma de protocolos de tráfico de Internet para el escenario 1.

Como se muestra en el gráfico 4.21 podemos determinar que ha existido una considerable cantidad de transferencia de datos en el rango del 0 a 8 [KBps], lo cual es normal porque es el valor correspondiente a las conexiones que se realizan al navegar por Internet.

El valor de densidad de frecuencia en el rango de 16-24[KBps] es aproximadamente la mitad del valor del rango de 0-8[KBps] y de la misma manera corresponde a conexiones que se realizan al navegar por Internet.

Se observa que el valor en el rango de 40-48[KBps] es aproximadamente similar al valor del rango de 0-8[KBps], además se puede deducir que este valor de densidad de frecuencia corresponde a descargas de datos pequeñas o páginas web que contengas videos, fotos de considerable tamaño.

Un dato relevante son los valores de los rangos de 88-96[KBps], 112-120[KBps], 120-128[KBps], 128-136 [KBps], estos datos corresponden a rangos altos de tasa de transferencia y tienen una considerable densidad de frecuencia por lo que se puede concluir que pertenecen a descargas de archivos.

Para confirmar la deducción anterior se visualiza el gráfico 4.22; este muestra que para el valor de del rango de 40-48[KBps] corresponde el 30% de los valores de bitrate calculados del tráfico de Internet, y el rango de 112-136[KBps] corresponde el 30% del mismo tráfico.

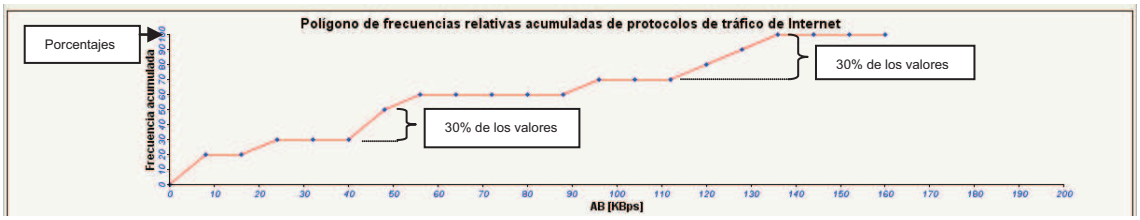


Gráfico 4.22 Polígono de frecuencias relativas acumuladas de protocolos de tráfico de Internet para el escenario 1.

Estos valores determinan que el 60% del tráfico total corresponden a descargas de datos, esto significaría una anomalía, dependiendo de las políticas de la empresa.

Tabla de Frecuencias

Intervalo de Clase (KBps)	Frecuencia	Frecuencia relativa	Densidad
0.0-<8.0	2	0,2	0,025
8.0-<16.0	0	0	0
16.0-<24.0	1	0,1	0,0125
24.0-<32.0	0	0	0
32.0-<40.0	0	0	0
40.0-<48.0	2	0,2	0,025
48.0-<56.0	1	0,1	0,0125
56.0-<64.0	0	0	0
64.0-<72.0	0	0	0
72.0-<80.0	0	0	0
80.0-<88.0	0	0	0
88.0-<96.0	1	0,1	0,0125
96.0-<104.0	0	0	0
104.0-<112.0	0	0	0
112.0-<120.0	1	0,1	0,0125
120.0-<128.0	1	0,1	0,0125
128.0-<136.0	1	0,1	0,0125
136.0-<144.0	0	0	0
144.0-<152.0	0	0	0
152.0-<160.0	0	0	0

Gráfico 4.23 Tabla de frecuencias para el escenario 1.

El gráfico 4.23 muestra los valores de frecuencia del gráfico 4.21 con mayor detalle para los valores de densidad de frecuencia.

Tabla de Estadística Descriptiva

Variable	N	Media	Error estándar de la media	Media recortada (5%)	Desviación estándar
Bitrate	10	62,1087	15,8301	62,1087	50,0593

Variable	Valor mínimo	Q1	Mediana	Q3	Valor máximo
Bitrate	0	12,502	49,9081	118,971	132,052

Gráfico 4.24 Tabla de estadística descriptiva para el escenario 1.

El gráfico 4.24 muestra que los datos no se han concentrado en la media sino que existen varios valores dispersos.

El administrador de red deberá implementar la solución pertinente por medio de herramientas de control de red especializadas para este escenario.

4.4.2. Segundo Escenario –Detección de posibles envíos de spam[10] o replicación de gusanos[11] informáticos.

Con los datos obtenidos del monitoreo de tráfico se requiere conocer las tasas de transferencia de conexión a Internet de las estaciones de trabajo a fin de localizar anomalías o comportamientos inusuales.

Las consecuencias de un ataque informático pueden ser visibles cuando una estación de trabajo infectada esté siendo utilizada, ya sea para enviar spam o por un gusano informático de red que se esté reproduciendo y enviando copias de sí mismo.

Para este tipo de análisis se requiere utilizar la herramienta “Tasa de transferencia promedio de uso de Internet “.

---

[10] Spam: Correo electrónico no deseado.

[11] Gusano informático: Es un tipo de software que tiene como objetivo infiltrarse o dañar una computadora sin el consentimiento de su propietario.

Se asigna la fecha y las horas en donde existió utilización de acceso a Internet para todas las estaciones de trabajo.

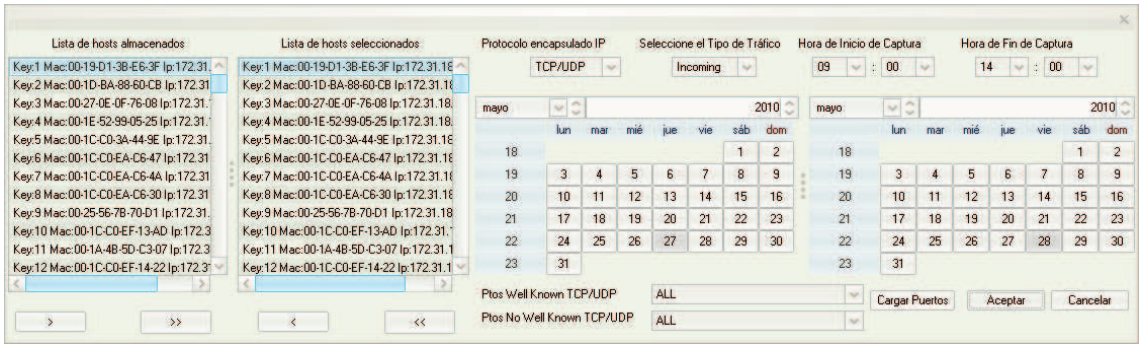


Gráfico 4.25 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos incoming (Tasa de transferencia promedio de uso de Internet).

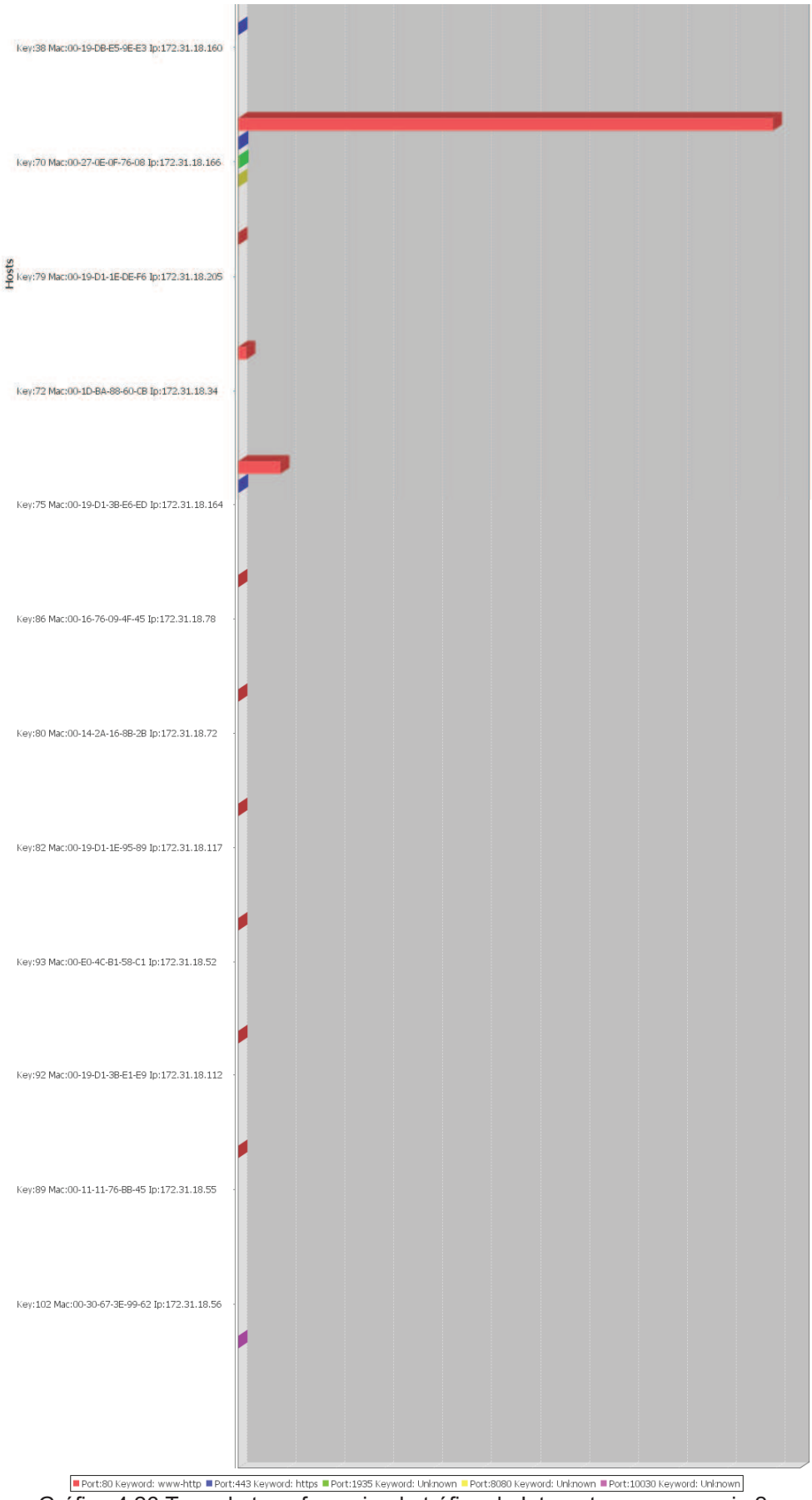


Gráfico 4.26 Tasa de transferencia de tráfico de Internet para escenario 2.

Como se muestra en el gráfico 4.26 existen 5 estaciones de trabajo con altos valores de tasa de transferencia de datos recibidos por el puerto 80. Las direcciones IPs correspondientes a estos hosts son:

- 172.31.18.159 - 172.31.18.58 - 172.31.18.161 - 172.31.18.166 - 172.31.18.164

Este tráfico puede ser ocasionado por descargas de archivos, en consecuencia se revisará el tráfico de salida para ver si las estaciones de trabajo antes señaladas poseen cantidades de tasa de transferencia considerables y descartar o afirmar una posible infección.

Para ello se asigna el filtro con los mismos valores de la anterior consulta a excepción del tipo de tráfico que en este caso será outgoing o saliente.

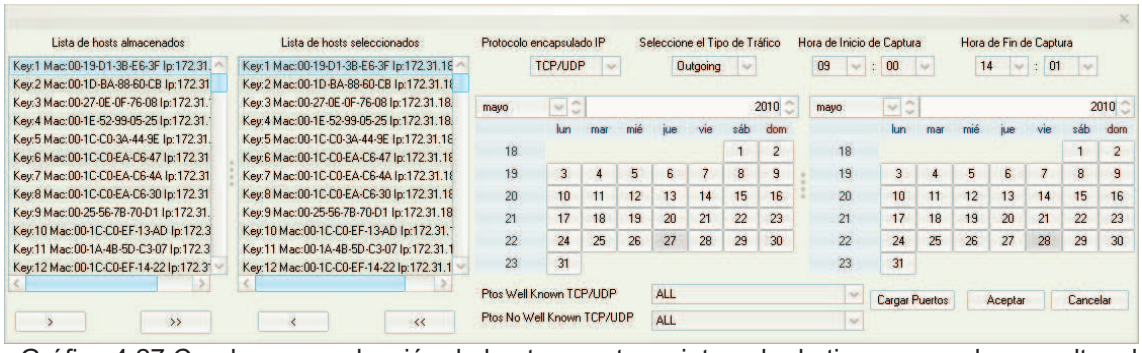
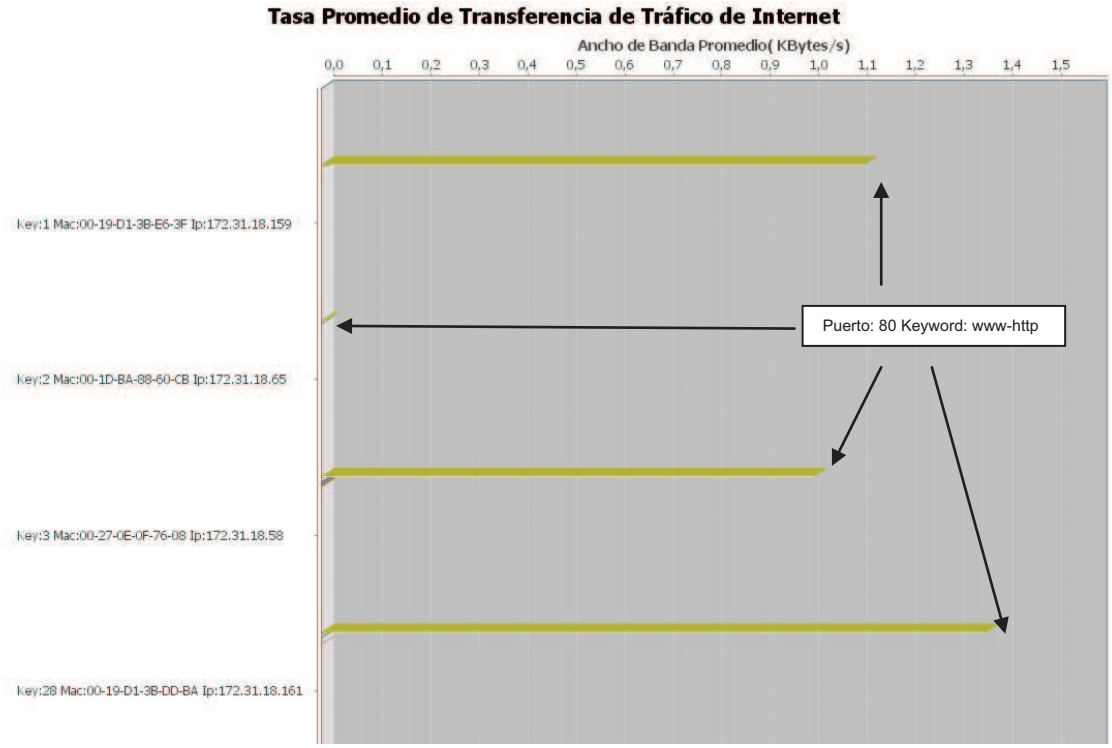


Gráfico 4.27 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos outgoing (Tasa de transferencia promedio de uso de Internet).



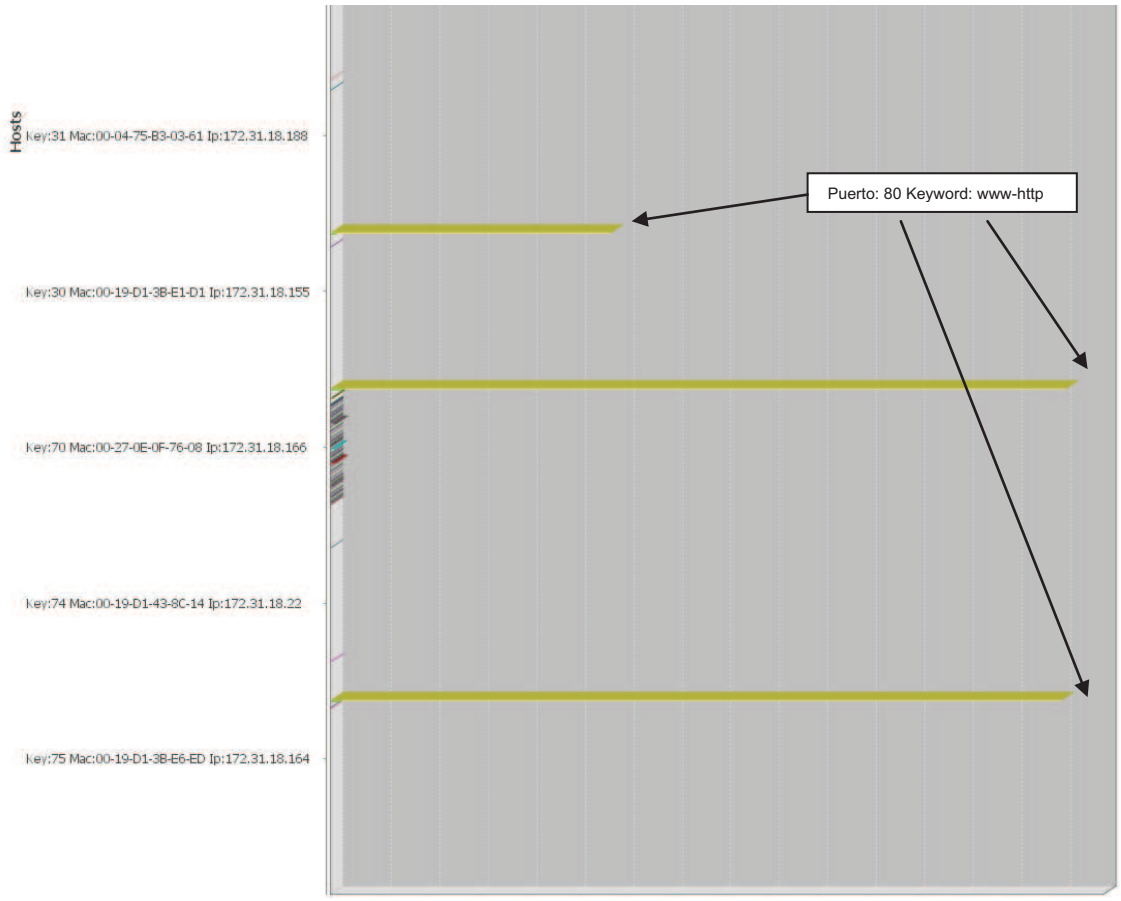


Gráfico 4.28 Tasa promedio de transferencia de tráfico de Internet para escenario 2.

Como se pueden apreciar en el gráfico 4.28 las estaciones de trabajo con mayor tasa de transferencia de salida son:

- 172.31.18.159 - 172.31.18.58 - 172.31.18.161 - 172.31.18.166 - 172.31.18.164 - 172.31.18.155

Los valores de bitrate para estas estaciones de trabajo son relativamente bajos por lo que se concluye que estos hosts no han sido infectados ni propagan la infección a otras máquinas.

En el caso de que los valores de bitrate sean altos, el administrador de red debe realizar la verificación de cada uno de los equipos y cerciorarse del uso que le dan los usuarios.

Por ejemplo, si se trata de una estación de trabajo que usa el correo electrónico para enviar notificaciones, informes y reportes de las actividades de la empresa, una tasa de transferencia de salida alta no es un caso inusual, pero si es una estación de trabajo sin estos privilegios el administrador debe tomar las acciones necesarias para corregir este problema.

4.4.3. Tercer Escenario – Direcciones IP sospechosas.

A partir de los datos obtenidos del monitoreo de tráfico de Internet se requiere saber si existe algún comportamiento inusual de una o de varias estaciones de trabajo. Siendo un comportamiento sospecho iniciar una conexión repetitiva a una dirección IP. Este tipo de anomalía describiría un posible ataque, ya que estaría conectándose con la estación de trabajo remotamente infectada para así causar daño en la red local. Para este tipo de análisis se requiere utilizar la herramienta "DNS reverso y ranking para IPs más utilizadas".

Se asigna la fecha y las horas en donde existió utilización del acceso a Internet para todas las estaciones de trabajo.

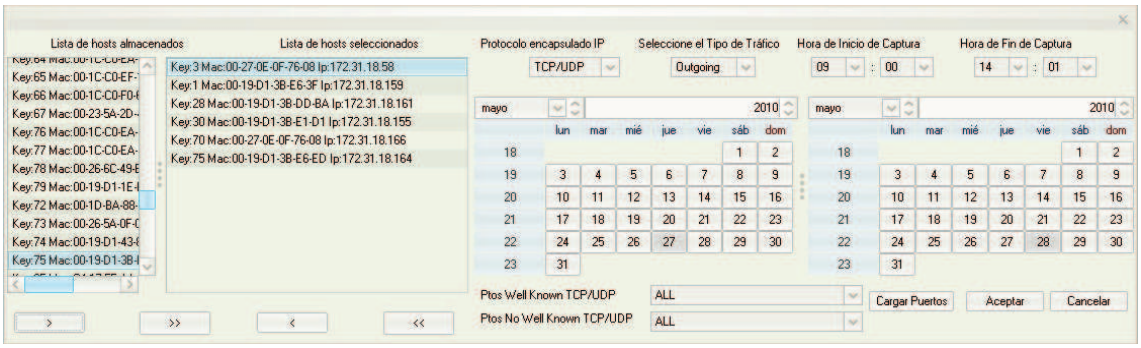


Gráfico 4.29 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos outgoing (DNS reverso y ranking para IPs más utilizadas).

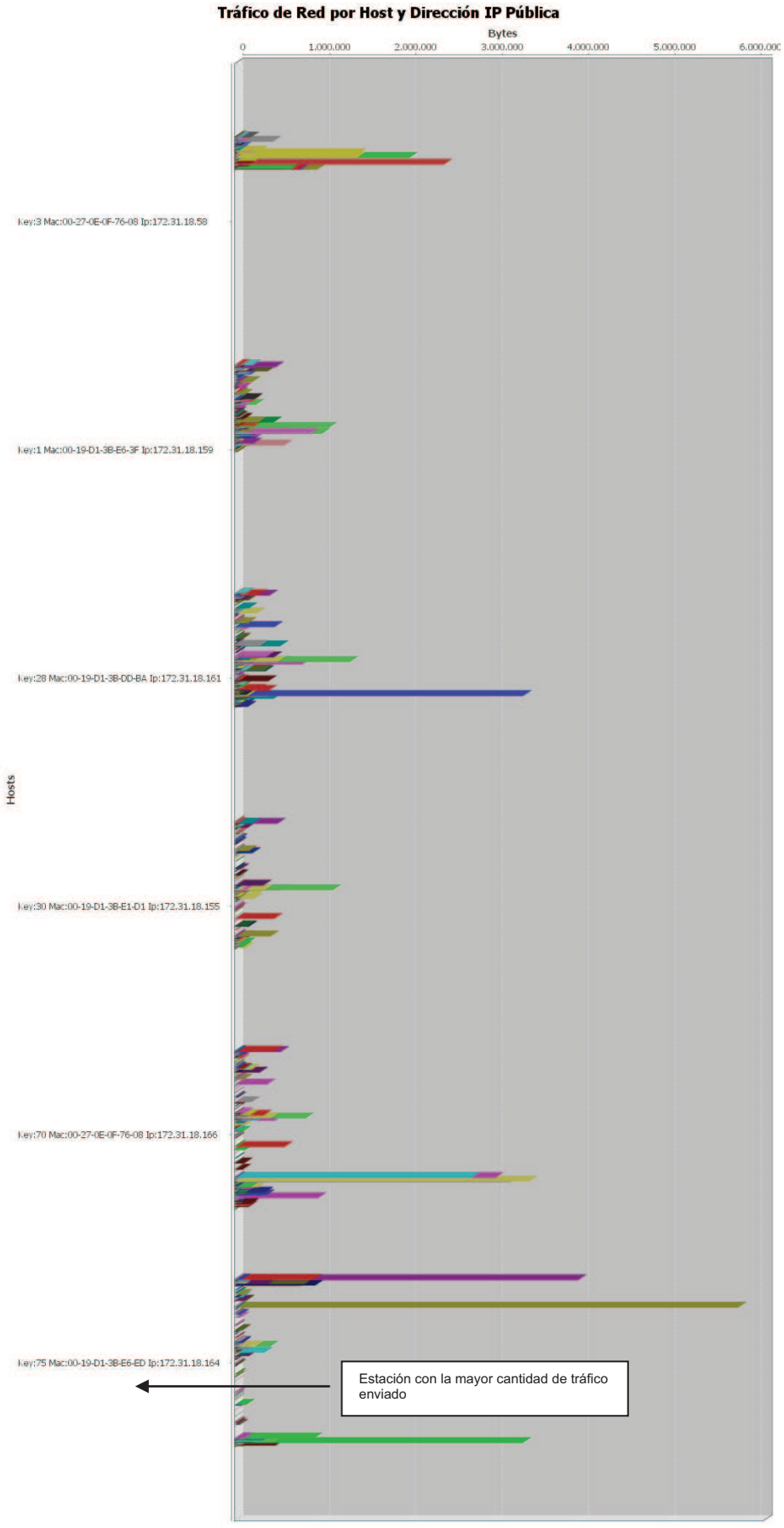


Gráfico 4.30 Tráfico de red por host y dirección IP pública para escenario 3.

Como se puede ver en el gráfico 4.30 la mayor cantidad de información transferida corresponde a la estación de trabajo con dirección IP 172.31.18.164 que oscila entre los 6 MB. Por ello se seleccionará esta estación de trabajo y se descartaría un posible ataque ya que la cantidad de datos enviados es mínima.

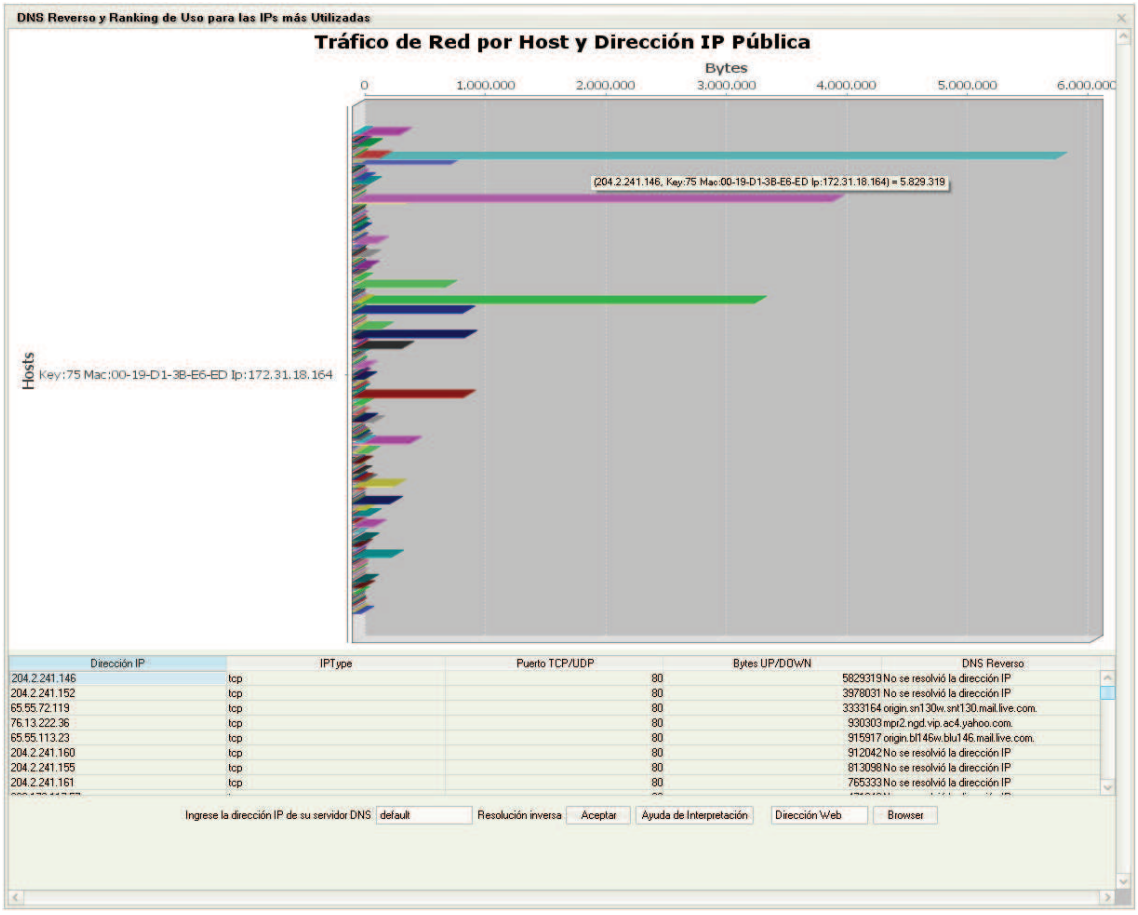


Gráfico 4.31 Tráfico de red por host y dirección IP pública con tabla de IP Ranking para el escenario 3.

En el gráfico 4.31 se observa que no se resuelven todas las direcciones IP ya que no existen registros de estas direcciones en el servidor DNS.

El administrador de red debería bloquear el acceso a direcciones IP que correspondan a tráfico no deseado de acuerdo a las políticas de la empresa.

4.4.4. Cuarto Escenario – Detección de patrones de comportamiento.

Con los datos obtenidos del monitoreo de tráfico de Internet se requiere saber si existe algún comportamiento inusual de la estación de trabajo con dirección IP 172.31.18.159 donde las descargas de grandes archivos no están permitidas. El objetivo es verificar si existe un patrón de conexiones repetitivas en horas y

periodos regulares del día. Para este tipo de análisis se requiere utilizar la herramienta “Series de tiempo en pasos“.

Se asigna la fecha y las horas en donde existió la utilización del servicio de Internet para la estación de trabajo con dirección IP 172.31.18.159.

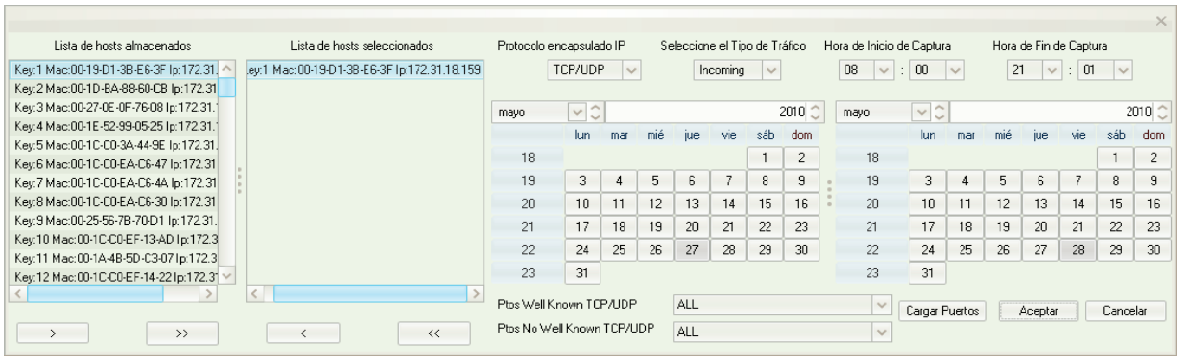


Gráfico 4.32 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos incoming (Series de tiempo en pasos)

Por ello se analizará la tasa de transferencia de este host en un gráfico de serie de tiempo durante los dos días de monitoreo.

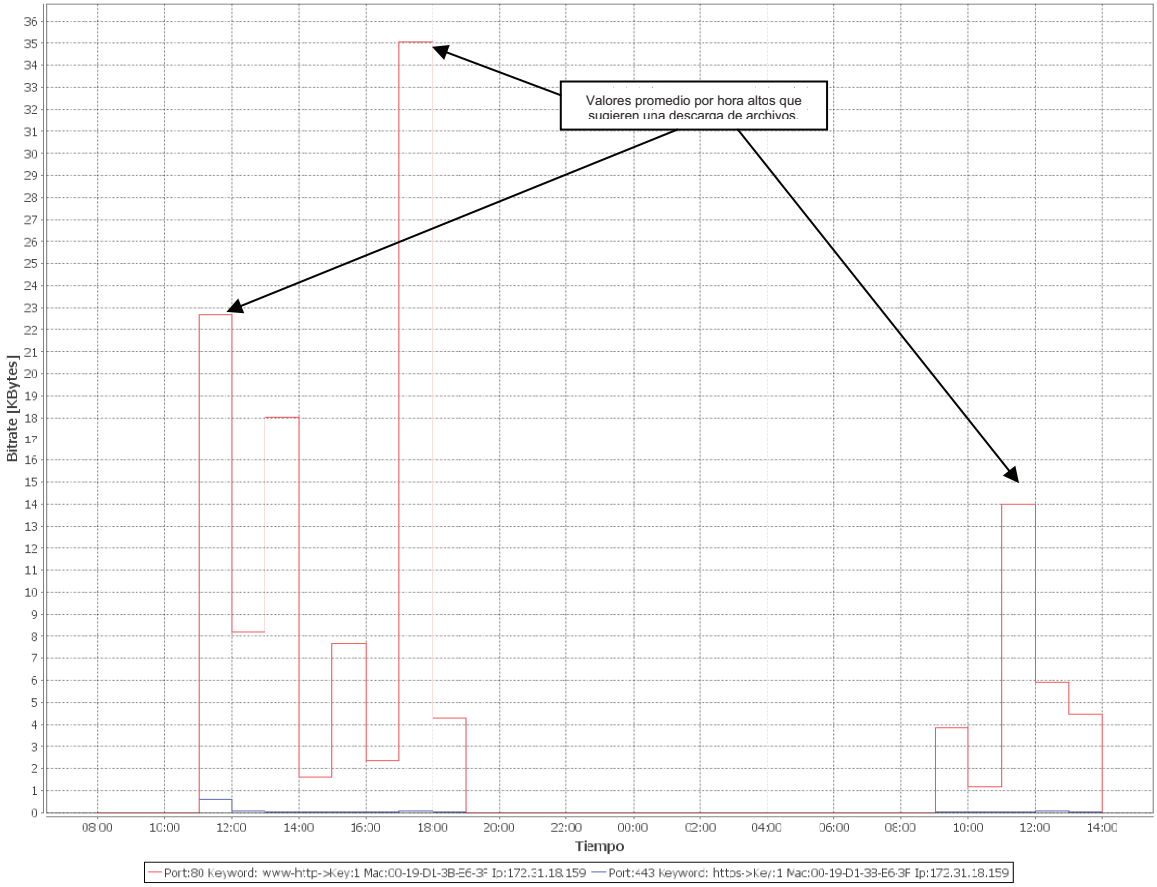


Gráfico 4.33 Tráfico Series de tiempo en pasos para el escenario 4.

Como se mostró en el gráfico 4.33 en el día jueves y viernes en el lapso comprendido entre las 11h:00 y 14h:00 existe mayor actividad correspondiente al puerto 80 de la estación de trabajo del ejemplo. Esto demuestra que existe un patrón de comportamiento común en ambos días del monitoreo que resulta en una actividad ilícita de descargas no autorizadas.

El administrador de red debe aplicar el reglamento interno de la institución o empresa para tomar las respectivas acciones correctivas.

4.5. ANÁLISIS DE COSTOS DE DESARROLLO

Aquí se detallarán los costos involucrados para el desarrollo del proyecto.

Para una mejor comprensión de los elementos involucrados se los ha subdivido en los siguientes puntos:

- Costos del diseño e implementación.
- Costo de equipos activos para pruebas.
- Costo de software.
- Costos indirectos.

4.5.1. Costos de diseño e implementación

Para los costos de diseño e implementación se necesitaron 2 programadores trabajando 8 horas diarias y 5 días a la semana por alrededor de 11 meses y medio.

El costo para un programador promedio con conocimientos en Java, MySQL, sistemas operativos, animación flash, conocimientos de básicos de Estadísticas y redes de datos está alrededor de 7.50 dólares americanos por hora. Este dato se obtuvo realizando una consulta del salario promedio para programadores en java en la conocida página web “multitrabajos”[12].

---

[12] http://www.bumeran.com.ec/ 2010-06-03

A continuación el cálculo del costo monetario de los programadores:

Costo Programador = ((7.5 dólares americanos x 8 horas x 20 días al mes)) x 11 meses + (7.5 dólares americanos x 8 horas x 10 días).

Costo Programadores Total = 13800 dólares americanos x 2 programadores.

Costo Programadores Total (Desarrollo del software)= 26400 dólares americanos.

- **Costo de equipos activos para pruebas**

En este tipo de costo se incluyen los equipos necesarios para la realización de pruebas durante el desarrollo, con la finalidad de depurar errores y comprobar el correcto funcionamiento del software.

Ítem	Descripción	Valor (dólares americanos)
Computadora escritorio	Procesador: 2.66 GHz Core2duo, Memoria RAM: 3 GB Disco Duro: 500 GB Interfaces de Red: 2 dispositivos	$ 800
Laptop Sony Vaio	Procesador: 2.3 GHz Core2duo, Memoria RAM: 3 GB Disco Duro: 250 GB Interfaces de Red: 1 dispositivos Sistema Operativo Windows 7	$ 950
Impresora	Impresora Laser Monocromática Hp Laserjet P2035n	$419
Switch D-link	Ethernet , No administrable, 8 puertos , 100 MBps.	$ 20
Hub	Ethernet ,8 Puertos , 10 MBps	$ 25
Patch Cords	2 metros certificados, categoría 6, marca Qpcom, conectores rj45	$ 6 c/u x 10 unidades = $60
Total		$ 2274

Tabla 4.3 Costo de equipos activos para pruebas.

4.5.2. Costos de software

Para el desarrollo del proyecto se requirió los siguientes paquetes de software.

Ítem	Descripción	Valor (dólares americanos)
Sistema Operativo	Licencia Windows XP SP3 Professional OEM Original Español	$100
Netbeans 6.5	Software de desarrollo	Gratuito
Adobe Flash Player Active 10X	Software para necesario para usar aplicaciones	Gratuito
Java ™ SE Development Kit 6 update 5	Necesario para el software de desarrollo y aplicaciones	Gratuito
Microsoft Office Home and Student 2007	Ofimática	$149
Bibliotecas para desarrollo de software.	Mysql-Connector 5.1.7, Mysql-Connector MXJ 5.0.9, Jsc1.0, Substance, Jfreechart 1.0.13, Native Swing.	Gratuito
Mozilla Firefox 3.0.19	Browser	Gratuito
Total		$249

Tabla 4.4 Costos de Software.

4.5.3. Costos Indirectos

A continuación se detallan los servicios básicos que fueron utilizados.

Servicio	Ítem	Valor(dólares americanos)
Electricidad	Requerido para los equipos	$22
Agua	Servicio básico	$27
Teléfono	Requerido para la conexión a Internet	$6.25 x 1.12 (pensión básica residencial)
Internet	Fast Boy (Banda Ancha)(512KBps)	$25 x 1.12
Imprevistos	Imprevistos	$120
Total		$204

Tabla 4.5 Costos Indirectos

4.5.4. Costo Total

El costo total resultante para la creación del software es:

Costo	Valor (dólares americanos)
Programadores	$ 26400
Equipos Activos	$ 2274
Software	$ 249
Indirectos	$ 204 x 11(meses)=$2244
Total	$ 31167

Tabla 4.6 Costo total del software

Hay que aclarar que este es el costo de la implementación, mas no es el precio de venta al público. Para tener una referencia del precio de venta al público se estiman una acogida similar al producto Colasoft Capsa 7.1 que posee más 5000 clientes alrededor de 80 países[13]. De acuerdo a este dato se calculará el precio de producción para el mínimo de clientes que adquirieron el software Colasoft Capsa a un precio de $549 dólares americanos por licencia (una estación de trabajo)[14].

Precio de venta al público referencial

PVP (Proyecto de Titulación) = 31167 dólares americanos / 5000 clientes potenciales.

Nota: Los clientes potenciales corresponden a la cantidad de clientes que la empresa Colasoft citada anteriormente tiene a nivel mundial.

PVP (Proyecto de Titulación) = 6.25 dólares americanos por licencia (una estación de trabajo)

Por lo tanto:

Producto	Valor precio por licencia (dólares americanos) (Una estación de trabajo)
Proyecto de titulación	$ 6.25
Colasoft Capsa 7.1	$ 549
Wireshark	$ 0

Tabla 4.7 Precio de venta al público

Este sería el precio y la cantidad de clientes necesarios para recuperar la inversión y el desarrollo del proyecto. Un análisis más profundo sobre este aspecto está fuera del alcance y los objetivos del proyecto.

[13] http://www.colasoft.com/company/about.php Visitado: 2010-06-24

[14] http://www.colasoft.com/purchase/capsaentprice.php Visitado: 2010-06-24

CAPÍTULO 5

5. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

- Mediante la utilización del histograma de frecuencias y el resumen de Estadística Descriptiva para el análisis de los datos capturados y almacenados en la base de datos, se hace evidente la naturaleza a ráfagas del tráfico de Internet ya que los instantes en que se envían y reciben los datos son impredecibles y el tamaño de estos bloques de información es variable. El principal indicador es la desviación estándar ya que indica cuan constante han sido los valores promedio de tasa de transferencia. Si el tráfico fuera invariable, el valor de la desviación estándar tendería a cero y las barras del histograma se reducirían a una sola que corresponde al intervalo de dicho valor.

- La diferenciación de tráfico de Internet es una característica importante para un administrador de red ya que a través de ella puede acceder a información más detallada y obtener criterios más acertados sobre el comportamiento de la intranet (particularmente de la conexión a Internet). Esto permitirá tener un mejor control sobre este servicio.

- Java como lenguaje de programación orientado a objetos incluye las facilidades para desarrollar soluciones de software basadas en conceptos estadísticos, que son distribuidas en librerías especializadas de acceso gratuito y de alcance didáctico y empresarial. Se suma además la potencialidad de la orientación a objetos por su carácter intuitivo, modular y reutilizable.

- JnetPcap es una gran herramienta para la programación de aplicaciones orientadas a redes de comunicaciones basadas en Ethernet, pues por medio de su API es posible utilizar desde el entorno de Java todas las funcionalidades de WinPcap (Windows) y libpcap (Unix) para la captura, establecimiento de filtros y decodificación de paquetes de una manera mucho

más fácil y productiva, usando para ello un lenguaje de programación orientado a objetos.

- JFreeChart facilita la generación de gráficos de gran calidad informativa y estética. La gratuidad de este paquete permite desarrollar aplicaciones en una fracción del tiempo que tomaría implementar los gráficos por cuenta propia de los programadores. Esto claro está, una vez que se entienda la estructura de clases del API descrito en su javadoc.

- Para desarrollar los métodos y funcionalidades relativas al cálculo de bitrate, las opciones para los filtros del sniffer (protocolos y puertos), organización de los paquetes decodificados tomando en cuenta los campos de las cabeceras de protocolo es necesario aplicar los conocimientos relativos a protocolos de redes de comunicaciones (redes LAN y TCP/IP) y teoría de la información de manera conjunta con los conocimientos y herramientas de la programación orientada a objetos. Sin un respaldo firme de conocimientos sobre estos aspectos claves de las redes de información, resulta extremadamente difícil para un programador sin instrucción en estas áreas, desarrollar aplicaciones como el actual proyecto de titulación.

- La resolución de problemas no previstos en el momento de diseño del desarrollo de software requiere investigar sobre la marcha en busca de mecanismos que permitan resolver eficientemente dicho problema. Este fue el caso del requerimiento de resolución inversa de nombres, donde fue necesario investigar más a fondo sobre el servicio de resolución de nombres de dominio DNS a través de implementaciones incluidas en el API de Java.

- Los resultados obtenidos en las pruebas del proyecto realizadas en el Laboratorio de Software, demostraron la estabilidad de la aplicación desarrollada para el proceso de captura de paquetes y almacenamiento en la base de datos interna. De la misma forma se verificó el proceso de recuperación de estos datos para la generación de los gráficos y resúmenes de datos.

- Se concluye que el manejo a fondo del IDE (*Integrated development environment*) de desarrollo, contribuye en gran medida a mejorar y depurar el código fuente, para hacerlo más eficiente tras la detección y corrección de errores.

- La realización de pruebas constantes y reajustes en el código fuente durante la implementación de funcionalidades específicas, permite optimizar y depurar errores antes de integrar estas pequeñas secciones al proyecto principal. De esta manera se evita la depuración de todo el código fuente ahorrando tiempo y disminuyendo la complejidad en el desarrollo del proyecto.

- Existen aplicaciones gratuitas y comerciales enfocadas al análisis de paquetes de red e incluso muy complejas que ofrecen un abanico impresionante de opciones avanzadas que deben estar acompañadas del conocimiento necesario por el usuario del programa. A diferencia de estos el presente proyecto realiza el análisis de los paquetes capturados orientado exclusivamente a obtener resúmenes y gráficos descriptivos que ayuden de manera visual e intuitiva a un administrador de red con conocimientos básicos en la toma de decisiones restrictivas o correctivas frente a un problema particular o uso inadecuado del servicio de Internet.

- El programar en un equipo de dos personas con una sola computadora permite obtener mejores resultados y un código de mejor calidad al desarrollar cualquier tipo de software. El código se vuelve más robusto al ser discutido y revisado en conjunto para detectar errores.

- Mantener un representante del cliente colaborando conjuntamente con el equipo de desarrollo, garantiza evaluar los requerimientos del software con más precisión, ayudando a identificar las prioridades fundamentales y en consecuencia disminuir el tiempo y costo de desarrollo, así como la cantidad de documentación.

- Para obtener un mejor control y adaptabilidad a cambios de requisitos por parte del cliente, es necesario realizar continuas pruebas en pequeños módulos funcionales que se mejoran paulatinamente y de forma incremental antes de anexarlos al proyecto principal, depurando todos los errores existentes previos a la implementación de una nueva funcionalidad.

5.2. RECOMENDACIONES

- Se recomienda a los profesionales de las redes de información incursionar en el campo de la programación para desarrollar aplicaciones personalizadas que resuelvan inconvenientes propios de la red administrada, pues la mayoría de programas existentes son enfocados a problemas de carácter general.

- El uso de herramientas de monitoreo y administración de tráfico de red es muy útil, siempre y cuando el administrador no las subutilice y se preocupe por capacitarse continuamente en la toma decisiones preventivas y correctivas, con el objetivo de optimizar el uso de los recursos disponibles.

- Se recomienda a todos los desarrollares de software que antes de usar una biblioteca especializada se lea cuidadosamente la documentación existente para determinar el alcance y la utilidad que aportaría al desarrollo de la aplicación para no subutilizar su capacidad.

- Se recomienda al programador ser paciente y persistente para alcanzar los objetivos planteados en cualquier proyecto de desarrollo de software. Esto es válido también como principio fundamental para tener éxito profesional.

- Se recomienda al programador tener entereza frente a los problemas no planificados que surgen durante la implementación de cualquier proyecto de desarrollo de software.

- Se recomienda tomar en cuenta todas las ideas sugeridas en el proceso de diseño e implementación, ya que la mejor solución puede surgir a partir de cualquiera de estas.

- Se recomienda a los estudiantes que deseen continuar con este proyecto de titulación, trabajar en la mejora de la apariencia y la amigabilidad del software con herramientas como adobe flash. La cual por su constante evolución permite mejorar la representación gráfica de los diagramas obtenidos a partir de los valores tratados.
- Para futuras mejoras el almacenamiento de datos puede utilizar una técnica más avanzada y compleja como es Java Transaction API, la cual permite tener un manejador de transacciones interactuando con JDBC y la aplicación de manera recursiva.
- Se recomienda poner un campo del tipo nombre en la base de datos que permita identificar las IPs de los hosts monitoreados que acceden a Internet. Debe realizarse un previo estudio que permita mostrar un óptimo rendimiento al realizar las consultas con grandes volúmenes de datos al implementar este identificador.
- Para aumentar la funcionalidad del programa se recomienda verificar la viabilidad de integrar los módulos de captura y análisis, sin afectar al rendimiento y la estabilidad de los mismos.

REFERENCIAS BIBLIOGRÁFICAS

Libros

1. BARCLAY, K; SAVAGE, J; **Objetc-Oriented Design with UML and Java**. Primera Edición. Editorial Elsevier Butterworth-Heinemann. Gran Bretaña. 2004.
2. CAPA SANTOS, Holger; **Modelación de Series Temporales**. Primera Edición. Escuela Politécnica Nacional. Quito. 2007.
3. DEITEL, Harvey M.; DEITEL, Paul J; **Cómo programar en Java**. Quinta Edición. Editorial Pearson.
4. DELAP, Scott; **Desktop Java Live**. Primera Edición. Editorial SourceBeat. Colorado. 2005.
5. NAVIDI, William; **Estadística para ingenieros**; Traducción de la Primera Edición. Editorial McGraw-Hill. México. 2006.
6. MySQL; **Manual de referencia de MySQL 5.0.** Revisión 357. 30-11-2006.
7. MySQL; **MySQL Connector/MXJ**. Revision 15704. 2009-07-16.
8. Object Management Group™ (OMG™); OMG Unified Modeling LanguageTM (OMG UML), Superstructure. Version 2.2. 2009-02-22.
9. SOMMERVILLE, Ian; **Ingeniería del software**. Séptima edición. Editorial Addison Wesley. España. 2005.
10. Spiegel, Murray R.; Stephens, Larry J.; **Estadística**. Tercera Edición. Editorial McGraw-Hill. México. 2002.
11. STALLINGS, William; **Comunicaciones y Redes de Computadores**. Sexta Edición. Prentice Hall.
12. STEVENS, Perdita; POOLEY, Rob; **Utilización de UML en Ingeniería del Software con Objetos y Componentes**. Traducción de la Primera Edición. Editorial Addison Wesley. España, 2002.
13. Sun Microsystems, Inc.; **Java Naming and Directory Interface™ - Service Provider Interface (JNDI SPI)**. v1.3. JNDI 1.2/Java™ 2 Platform. 1999.
14. Sun Microsystems, Inc.; **Java Naming and Directory Interface™ - Application Programming Interface (JNDI API)**. v1.3. JNDI 1.2/Java™ 2 Platform, Standard Edition. 1999.

Direcciones Electrónicas

1. http://www.winpcap.org/
   Desde: año 1999, Visitado: año 2009.
2. http://jnetpcap.com/
   Desde: año 2005, Visitado: año 2009.
3. http://www.ietf.org/
   Visitado: año 2009.
4. http://java.sun.com/jndi
   Visitado: año 2010.
5. http://java.sun.com/j2se/1.5.0/docs/guide/jndi/index.html
   Desde: año 2002, Visitado: año 2009.
6. http://java.sun.com/docs/books/tutorial/uiswing/components/index.html
   Desde: año 1995, Visitado: año 2009.
7. http://www.jsc.nildram.co.uk/
   Desde: año 2005, Visitado: año 2009.
8. http://dev.mysql.com/doc/refman/5.0/es/introduction.html
   Desde: año 1997, Visitado: año 2009.
9. http://www.jfree.org/jfreechart/
   Desde: año 2005, Visitado: año 2010.
10. http://djproject.sourceforge.net/ns/
    Desde: año 2009, Visitado: año 2010.
11. http://www.toedter.com/
    Desde: año 2009, Visitado: año 2010.
12. http://netbeans.org/
    Desde: año 2000, Visitado: año 2009.
13. http://www.omg.org/spec/UML/2.2/
    Desde: año 1997, Visitado: año 2009.
14. http://es.wikipedia.org/wiki/Domain_Name_System
    Desde: año 2010, Visitado: año 2010.
15. http://en.wikipedia.org/wiki/Reverse_DNS_lookup
    Desde: año 2010, Visitado: año 2010.
16. http://www.altova.com/umodel.html
    Desde: año 2005, Visitado: año 2010.

Documentación

Javadoc de JDK 1.6.0_05

Javadoc de jNetPcap 1.2.rc5

Javadoc de JSC 1.0

Javadoc de JFreeChart 1.0.13

Javadoc de NativeSwing (DJNativeSwing-SWT-0-9-9-20100221)

Javadoc de JCalendar v1.2.2

Documentación de WinPcap 4.0.2

ANEXOS

ANEXO A

A. INTRODUCIÓN A UML

UML es el acrónimo de Unified Modelling Language (Lenguaje Unificado de Modelado). En el presente proyecto se utiliza la versión 2.2 de UML. Como su nombre lo indica es un lenguaje de modelado, esto significa que sirve para describir un modelo de un diseño o sistema de manera suficientemente descriptiva, tomando en cuenta los aspectos esenciales y sin profundizar en detalles innecesarios.

El Lenguaje Unificado de Modelado no está atado a ningún método de desarrollo particular, de hecho UML puede ser utilizado por cualquier método OO (Orientado a Objetos).

Este lenguaje de modelado se centra en la elaboración de diagramas para visualizar, especificar, documentar y construir los elementos de un diseño orientado a objetos. Los diagramas utilizados para describir los elementos y la funcionalidad del presente proyecto de titulación son los casos de uso, los de clase y los de actividad.

A.1. Casos de Uso

Un caso de uso representa la interacción que existe entre un usuario y el sistema. Son usados para describir y documentar el comportamiento del sistema y para la captura de requisitos de software, es decir, lo que se espera que el sistema sea capaz de hacer.

La principal ventaja de los casos de uso es su fácil comprensión, proporciona un acercamiento al cliente y sirve como base para las pruebas a realizarse sobre el sistema en la etapa de desarrollo.

En caso de uso, los usuarios son descritos como un actor que cumple un rol específico. Hay que aclarar que un actor no necesariamente representa a una persona, muchas veces confundido por que su notación es la de un muñeco. Un actor puede ser cualquier cosa externa al sistema que interactúa con él; sea esto un dispositivo físico u otro sistema automatizado de software.

Un caso de uso, que se dibuja como un óvalo y su nombre, representa de manera simplificada una tarea a realizarse. La línea que une a un actor con un

caso de uso significa que este interactúa o participa en la realización del mismo. Un actor se simboliza por un muñeco con forma humana.

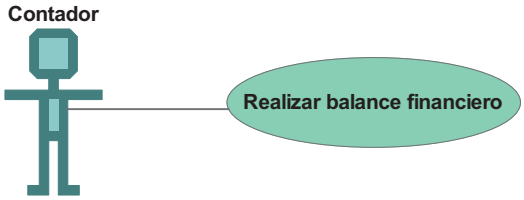


Gráfico A.1 Diagrama de casos uso simple

Existen ocasiones en las que es posible encontrar un comportamiento común en dos o más casos de uso, que luego podría ser implementado por separado para no redundar en funcionalidades semejantes. Esta característica se simboliza como se muestra en el siguiente gráfico.

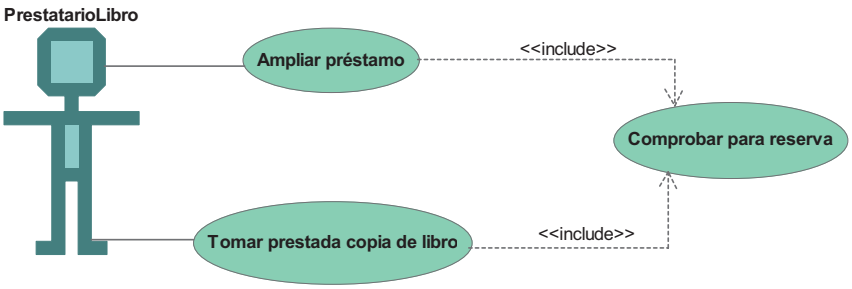


Gráfico A.2 Reutilización de casos de uso [1]

A.2. Diagramas de Clase

Los diagramas permiten describir la estructura estática de un sistema, con sus clases y relaciones. Un diagrama de clases incluye a la estructura de paquetes de clases.

La representación básica de una clase es de un rectángulo con un nombre.

Gráfico A.3 Representación de una clase sencilla

Una clase además consta de un conjunto de atributos y operaciones. El conjunto de atributos representan los datos contenidos dentro de un objeto de esta clase. Estos datos encapsulados en el período de vida de un objeto se los conoce como “estado”. Las operaciones corresponden a los mensajes que un

[1] Fuente: STEVENS, Perdita; POOLEY, Rob; Utilización de UML en Ingeniería del Software con Objetos y Componentes; Cap. 8 ; pág. 120

objeto de esta clase puede recibir. Las operaciones especifican un nombre, el tipo de retorno, los parámetros y restricciones para invocar un comportamiento asociado. Cuando una operación es invocada, se ejecuta un segmento de código que implementa la funcionalidad deseada; a este código se le conoce como un método.

Los atributos y las operaciones tienen niveles de visibilidad; esto determina el grado de accesibilidad por parte de otras clases a los elementos de una en particular. En el desarrollo se pueden especificar cuatro niveles de visibilidad.

Visibilidad	Símbolo UML	Símbolo equivalente Altova UModel Atributos/Operaciones	Descripción
public	+		Disponible para todas las clases
protected	#		Disponible para las subclases
private	-		Disponible solo para la clase en sí
package	~		Disponible para todas las clases en el mismo paquete

Tabla A.1 Visibilidad de los atributos y operaciones

Por lo general para mostrar las relaciones entre clases es más conveniente utilizar su representación más sencilla, sin embargo, cuando se requiere conocer al detalle la estructura de un sistema, es necesario describir apropiadamente sus clases con sus atributos y operaciones.

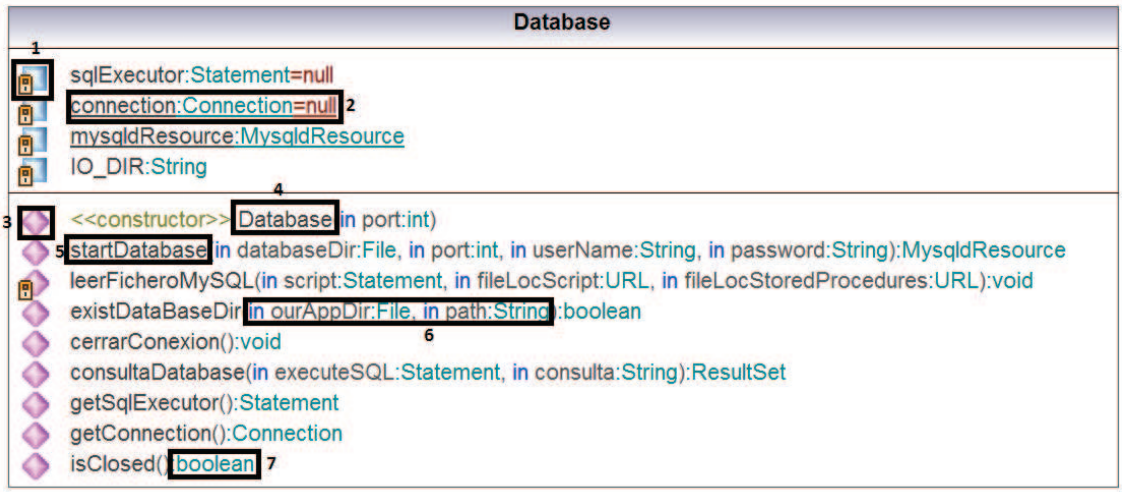


Gráfico A.4 Notación de un diagrama de clase

La siguiente tabla describe la notación indicada en el gráfico anterior.

Indicador	Descripción
1. Visibilidad del atributo	Símbolo que identifica cual es la visibilidad del atributo.
2. Atributo	En primer lugar se lee el nombre del atributo y separado por dos puntos (:) se especifica el tipo de dato, que puede ser un tipo simple o una clase. En este caso el atributo es estático; esto es indicado porque se encuentra subrayado.
3. Visibilidad de la operación	Símbolo que identifica cual es la visibilidad de la operación.
4. Constructor	Nombre del constructor de la clase. La primera letra se escribe con mayúscula.
5. Operación	Nombre de la operación. En general se escribe con minúsculas y si se combina con otras palabras, la primera de estas se escribe con mayúscula.
6. Parámetros	En el ejemplo, escrito en azul se encuentra el tipo de dirección del parámetro. Existen tres opciones: 1. in Indica que los valores del parámetro son ingresados por quien llama al métodos de esta operación. 2. inout Indica que los valores del parámetro son ingresados por quien llama al métodos de esta operación y enviados de vuelta a quién hizo la llamada. 3. out Indica que los valores del parámetro son pasados de una operación hacia quién hizo la llamada. Posterior a esto se encuentra el nombre del parámetro y separado con dos puntos (:) la especificación del tipo de parámetro, que puede ser un tipo básico o una clase.
7. Tipo de retorno	Separado con dos puntos (:) de la operación y los parámetros, se especifíca el tipo de retorno que puede ser un tipo de dato básico, una clase o ninguno (void).

Tabla A.2 Descripción de la notación de un diagrama de clase.

A.2.1. Asociaciones

Las asociaciones expresan las relaciones entre las clases. Los objetos de las clase asociadas cooperan entre ellos con el envío de mensajes el uno al otro.

“Sean dos clases A y B, están asociadas si:

-Un objeto de la clase A envía un mensaje a un objeto de la clase B.

```
public class A {

  ...

  public A(){

  ...

  }

  //Un objeto de la clase A envía un
  //mensaje a un objeto de la clase B
  //solicitando la ejecución del
  //metodoB1().
  public void metodoA1(){

          //b es un objeto de clase B
          b.metodoB1();
          ...
  }

}
```

Ejemplo de código A.1

-Un objeto de la clase A crea un objeto de la clase B.

```
public class A {

  ...

  public A(){

  ...

  }

  //Cuando un objeto de la clase A
  //ejecute el metodoA2() creará una
  //instancia de la clase B.
  public void metodoA2(){
        B b1 = new B();

        ...

  }
```

Ejemplo de código A.2

-Un objeto de la clase A tiene un atributo cuyos valores son objetos de la clase B o colecciones de objetos de la clase B.

```
public class A {

  //un objeto como atributo

  private B b1;

  ...

  public A(){

  ...

  }

  public void metodoA3(){

          ...

  }

}
```

Ejemplo de código A.3

-Un objeto de la clase A recibe un mensaje con un objeto de la clase B pasado como argumento."[2]

```
public class A {
   ...
   ...
   public A(){
   ...
   }
   public void metodoA4(B b){
           ...
   }
}
```

Ejemplo de código A.4

Gráfico A.5 Asociación sencilla entre clases

A.2.2. Navegabilidad

Para añadir navegabilidad a una asociación, se lo hace por medio de una flecha en uno o ambos extremos de la línea que la representa. La dirección de la flecha indica la factibilidad de enviar mensajes en esa dirección.

Gráfico A.6 Asociación de navegación en un solo sentido

No existe ningún inconveniente con que las asociaciones que posean navegabilidad incluyan multiplicidad.

```
public class A {
   //una colección de objetos como
   //atributo
   private ArrayList<B> listaB;
   ...
   public A(){
   ...
   }
   public void metodoA5(){
           ...
   }
```

Ejemplo de código A.5

[2] Fuente: STEVENS, Perdita; POOLEY, Rob; Utilización de UML en Ingeniería del Software con Objetos y Componentes; Cap. 5 ; pág. 73

En el ejemplo de código A.5 se dice que A *conoce* a B, pero no al revés. En este caso A puede enviar mensajes a los objetos instancia de la clase B de la colección y acceder a su información fácilmente.

A.2.3. Multiplicidades

La multiplicidad permite especificar cuantos objetos de una clase se asocian con la de la otra. Estos valores pueden ser:

"-Un número exacto, simplemente escribiéndolo.

-Un rango de números, utilizando dos puntos entre un par de números.

*-Un número arbitrario, no especificado, utilizando *."*[3]

Los siguientes ejemplos combinan navegabilidad y multiplicidad.

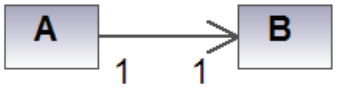


Gráfico A.7 Asociación uno a uno: por cada objeto de la clase A hay un objeto de la clase B asociada con A (ver ejemplo de código A.3).

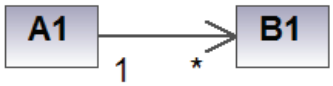


Gráfico A.8 Asociación uno a muchos: cada objeto de la clase A1 está asociado con ninguno o varios objetos de la clase B1 (ver ejemplo de código A.5).

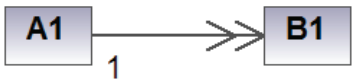


Gráfico A.9 Asociación de colección: Un objeto de clase A está asociado con una colección de objetos de la clase B, es equivalente al gráfico A.8 (ver ejemplo de código A.5).

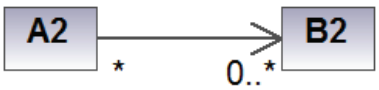


Gráfico A.10 Asociación muchos a muchos: cada objeto de la clase A2 está asociado con varios objetos de la clase B2 (ver ejemplo de código A.6).

[3] Fuente: STEVENS, Perdita; POOLEY, Rob; Utilización de UML en Ingeniería del Software con Objetos y Componentes; Cap. 5 ; pág. 75

```
//Primero un objeto de la A2 tiene una
//colección de objetos de la clase B2
//(uno a muchos).

public class A2 {

  //una colección de objetos B2

  private ArrayList<B2> listaB2;

  ...

  public A2(){...}

  ...

  ...

}

//Para implementar la asociación 'muchos
//a muchos', una tercera clase debe
//contener una colección de objetos de
//la clase A2. El atributo listaA2
//implementa la asociación 'muchos a
//muchos' entre las clases A2 y B2.

public class C {

//una colección de objetos A2

  private ArrayList<A2> listaA2;

  ...

  public C(){...}

  ...
```

Ejemplo de código A.6

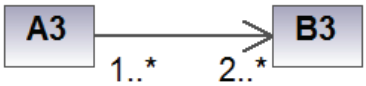


Gráfico A.11 Asociación uno o más a dos o más: cada objeto de la clase A3 está asociado con dos o más objetos de la clase B3 y cada objeto de la clase B3 está asociado con uno o más objetos de la clase A3.

El gráfico A.11 muestra un caso particular al del gráfico A.10 pero con restricciones de multiplicidad más precisas. Estas restricciones en general son parte de los requerimientos de lo que se quiere representar. Por ejemplo en una institución educativa se dictan cursos donde al menos **uno** siempre se encuentra activo y como mínimo deben tener **dos** estudiantes (ver ejemplo de código A.7).

```
//La clase Curso representa a un curso a dictarse
//en la institución educativa X.

public class Curso {

  //una colección de objetos Estudiante

  private ArrayList<Estudiante> listaEstudiantes;

  private String nombreCurso;

  private int duracionHoras;

  private String nombreInstructor;

  ...

  public Curso(){...}

  ...

  ...

}

//La clase AdministradorCursos se encarga de
//realizar las respectivas validaciones para que
//la multiplicidad entre la clase Curso y
//Estudiante sea la del requerimiento.

public class AdministradorCursos {

//una colección de objetos Curso

  private ArrayList<Curso> listaCursos;

  ...

  public AdministradorCursos(){...}

  ...

  ...

}
```

Ejemplo de código A.7

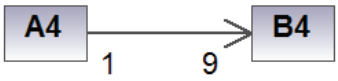


Gráfico A.12 Asociación de multiplicidad con valores fijos: por cada objeto de la clase A4 hay nueve objetos de la clase B4 que están asociados con A4 y cada objeto de la clase B4 está asociado con un objeto de la clase A4 (ver ejemplo de código A.8).

```
public class A4 {

  private B4[] = new B4[9];

  ...

  public A4(){...}

  ...
  ...
}
```

Ejemplo de código A.8

A.2.4. Generalización

Es la representación de clases con características generales y comunes a otras que pueden considerarse especializaciones o clases derivadas de las primeras. El siguiente ejemplo aclarará el punto anterior.

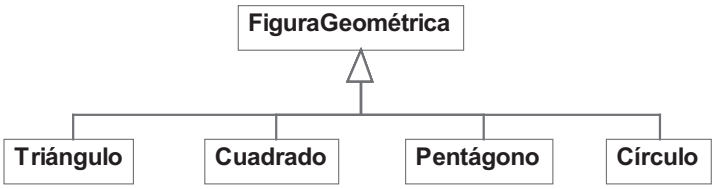


Gráfico A.13 Diagrama de clases que contiene generalización

El gráfico A.13 muestra que la clase FiguraGeometrica es una generalización de las clases Triángulo, Cuadrado, Pentágono y Círculo o que estas últimas son especializaciones de la primera. Una clase especializada contiene los atributos y operaciones de la "superclase o clase base" en este caso FiguraGeometrica y potencialmente agregue otros.

```
//Superclase
public class FiguraGeometrica{
  private Color color;
  ...
  public FiguraGeometrica(){...}
  public void borrar(){...}
  public void dibujar(){...}
  public Color getColor(){
        return color;
  }
  public void setColor(Color newColor){
        color = newColor;
  }
}
//Subclase
public class Cuadrado extends FiguraGeometrica {
  double lado;
  public Cuadrado(){
  super();
  ...
  }
  public void setLado(double newLado){
        lado = newLado;
  }
  public double calcularPerimetro(){
        return lado*4D;
  }
}
```

Ejemplo de código A.9

A.2.5. Agregación de composición

La agregación de composición es un tipo de asociación entre objetos, que denota que un objeto de una clase es parte de un objeto de otra clase. La relación entre objetos que tienen este tipo de asociación es mucho más fuerte que una normal, la totalidad no puede existir sin sus partes y las partes no pueden existir sin la totalidad. Esto último implica los siguientes puntos que se muestran a continuación:

-El borrado o copiado de la "totalidad" implica que sus partes se copian o se borran con él.

-Solo existe un objeto "todo", las partes no están compartidas con otros objetos "todo". Esto se traduce en que la multiplicidad en el extremo del "todo" debe ser 1.

-No se puede acceder a las partes afuera de la "totalidad", estas son privadas para el "todo" y un mensaje destinado para una parte debe ser enviado al "todo" y retransmitido por este a la parte.

"Esto significa que una agregación de composición sólo debería ser usada cuando un objeto es considerado parte de otro objeto y no solo una asociación casual con una existencia y visibilidad independiente."[4]

Hay que aclarar que el símbolo del diamante se coloca en el extremo del todo, no de la parte.

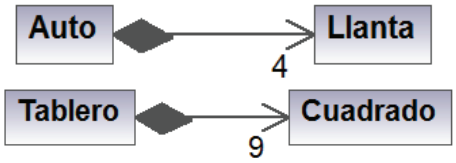


Gráfico A.14 Ejemplos de composición

Un ejemplo típico para una agregación es el caso del juego Tres en raya donde están involucradas las clases Cuadrado y Tablero. Se entiende que un Cuadrado es parte de un Tablero, y como consecuencia de estar relacionados de manera tan directa si se copia o borra un objeto Tablero, se copiarán o borrarán los objetos Cuadrado que son parte del Tablero.

[4] BARCLAY, K; SAVAGE, J; Objetc-Oriented Design with UML and Java. (1ª Edición); Cap. 2 ; pág. 42

```
public class Tablero {

  private Cuadrado[] = new Cuadrado[9];

  ...

  public Tablero(){...}

  ...

  ...

}
```

Ejemplo de código A.10

A.2.6. Agregación compartida

Una agregación compartida conserva el criterio de la parte y el “todo”, pero en este caso las partes son compartidas con otros “todo”. Al igual que la agregación de composición un mensaje destinado a la parte debe ser enviado a través del todo. Pero es este caso el borrado de un “todo” no implica que las partes se borren con él, estas partes pueden ser usadas por otros “todo” de manera independiente. La multiplicidad en el extremo del “todo” puede ser mayor a 1.

Al igual que en una composición el símbolo del diamante se coloca en el extremo del todo, no de la parte. El diamante en este caso está vacío.

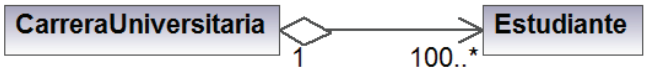


Gráfico A.15 Agregación compartida

```
public class FacultadElectronica {
  private ArrayList<CarreraUniversitaria > listaCarreras;
  private ArrayList<Estudiante> listaEstudiantesFacultad;
  public Facultad(){...}
  ...
}
public class CarreraUniversitaria {
  private ArrayList<Estudiante> listaEstudiantesCarrera;
  ...
  public Carrera(){...}
  ...
  ...
}
```

Ejemplo de código A.11

En el ejemplo de código A.11 se implementa la asociación de agregación compartida. Para entender mejor este tipo de asociación se propone el siguiente escenario:

Una clase FacultadElectronica administra las diferentes instancias de clase u objetos de CarreraUniversitaria, como por ejemplo ingeniería electrónica en control, telecomunicaciones y redes de información. Dado el caso que una de las carreras deba ser eliminada por insuficiente cantidad de estudiantes, no implica que los objetos estudiante deban ser borrados conjuntamente con la carrera, pues ellos pueden también formar parte de otra, por esta razón existe una colección de todos los estudiantes de la Facultad.

A.3. Diagramas de Actividad

Los diagramas de actividad permiten describir los aspectos dinámicos del sistema, es decir, como se coordinan las actividades para llevar a cabo una funcionalidad, es el flujo de control entre ellas.

Son usadas para documentar casos de uso y el comportamiento de los métodos de una clase.

Elementos de los diagramas de actividad:

- **Marcas de inicio y finalización**. Son símbolos que indican el inicio y el fin del flujo de actividades. Su notación se indica gráfico A.16.

- **Actividad**. Representadas por una caja con esquinas redondeadas que contienen texto que describe la acción realizada. Una actividad implicar varios pasos internos o la espera de eventos, aunque estos generalmente no se muestran.

- **Transición**. Representada por una flecha que indica la siguiente actividad en el flujo.

- **Barra de Sincronización**. Su notación gráfica es la de una barra gruesa a la llegan o parten transiciones. Son usadas para representar varias situaciones comunes con subtareas, como por ejemplo, la unión y la división.

La unión consiste en esperar a que varias actividades separadas culminen antes de continuar. La división consiste en iniciar varias actividades en paralelo.

- **Diamante de decisión**. Permite representar decisiones. Un diamante de decisión tiene una transición de entrada y dos o más transiciones de salida. Cada transición de salida contiene una etiqueta que identifica una alternativa de decisión.

En diagramas de actividad detallados es útil mostrar las acciones tomadas frente a la ocurrencia de un evento. Los eventos son quienes desencadenan la ejecución de una actividad. A continuación se describe el evento de tiempo, utilizado en este proyecto para mostrar la realización de actividades periódicamente.

- Evento de tiempo. Denota el tiempo que tiene que pasar después de un evento.

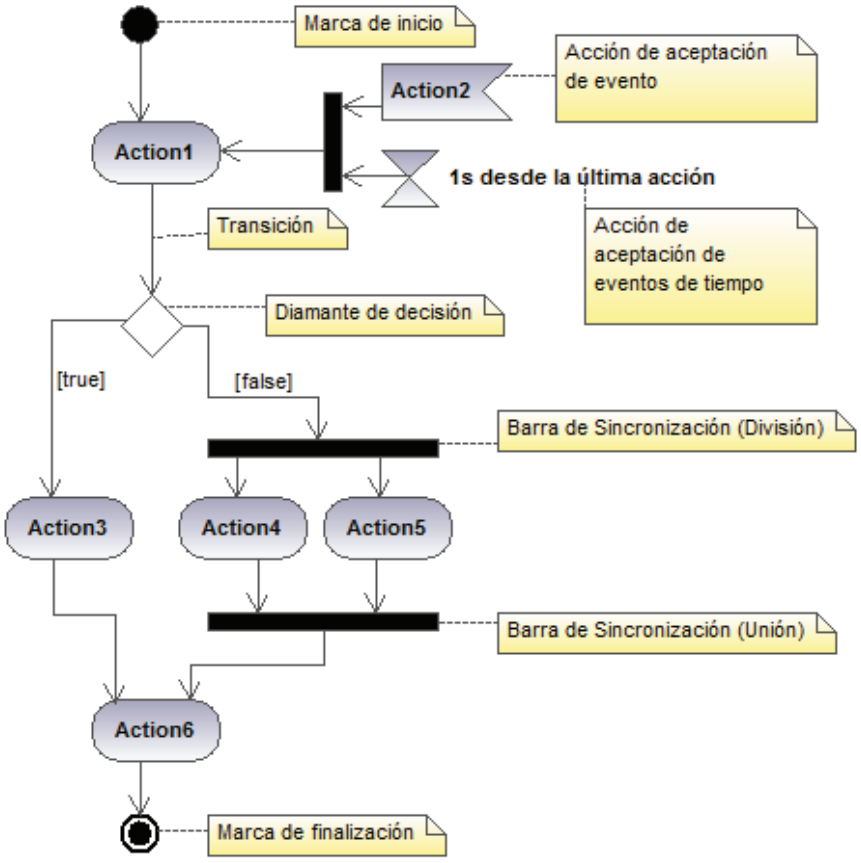


Gráfico A.16 Ejemplo ilustrativo de un diagrama de actividad

A.4. Diagramas de Interacción

Los diagramas de interacción permiten almacenar de manera gráfica la forma en que los objetos interactúan para realizar un caso de uso o un escenario particular de un caso de uso.

Existen dos tipos de diagramas de interacción, los diagramas de secuencia y los diagramas de colaboración. Ambos almacenan casi la misma información pero con un enfoque diferente.

Elementos comunes en los diagramas de interacción

Objetos. Cada uno aparece como un rectángulo, etiquetado de la forma *nombreObjeto:nombreClase.* Puede haber dos o más objetos diferentes de una misma clase.

Mensajes. Representados por una flecha etiquetada, representa a un mensaje enviado por el objeto que está en la cola de la flecha hacia el que apunta la flecha.

Actores. Estos diagramas pueden incluir actores como en un caso de uso. Pueden ser usados para definir el rol del usuario que estimula al sistema para comportarse en la forma descrita en el diagrama.

A.4.1. Diagramas de secuencia

Estos diagramas muestran la secuencia de mensajes que toman lugar entre los objetos en interacción, con énfasis en el orden temporal. El tiempo pasa según se desplaza de arriba hacia abajo.

El orden en que aparecen los objetos es arbitrario, pero se facilita la interpretación del mismo si los objetos que participan antes se ubican primeros de izquierda a derecha.

Incluye elementos propios tales como:

Línea de vida del objeto. Es la línea vertical que representa la vida del objeto durante el tiempo de la interacción.

Activaciones. Son pequeños rectángulos sobre las líneas de vida para aclarar cuando un objeto está envuelto en algún tipo de procesamiento, como la ejecución de un método.

Retorno. Flechas entrecortadas que representan la respuesta a un mensaje. Normalmente el retorno se considera implícito.

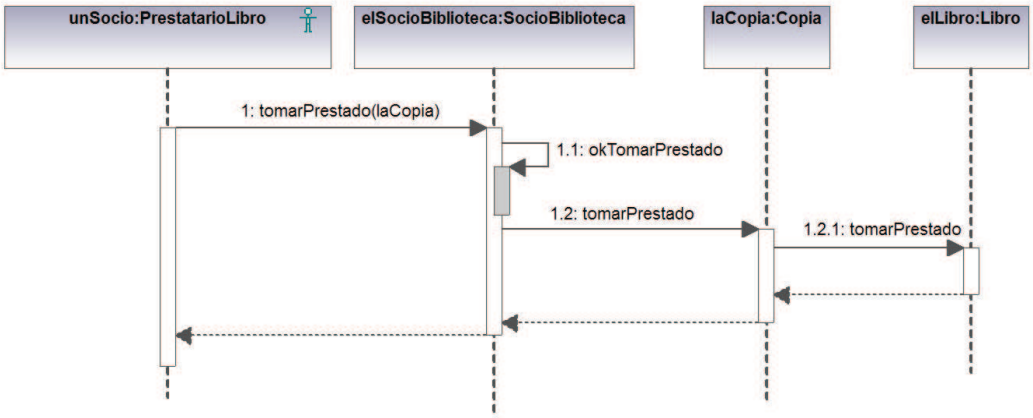


Gráfico A.17 Ejemplo ilustrativo de un diagrama de secuencia [5]

A.4.2. Diagramas de colaboración

Los diagramas de colaboración se enfocan en las relaciones estructurales que existen entre los objetos y los mensajes que son enviados.

Enlaces. Son iguales a las asociaciones en el modelo de clases. La colaboración no tiene que incluir enlaces para todas las asociaciones, sólo para las realmente importantes. Puede incluirse navegabilidad o el nombre del enlace si eso hace al diagrama más claro.

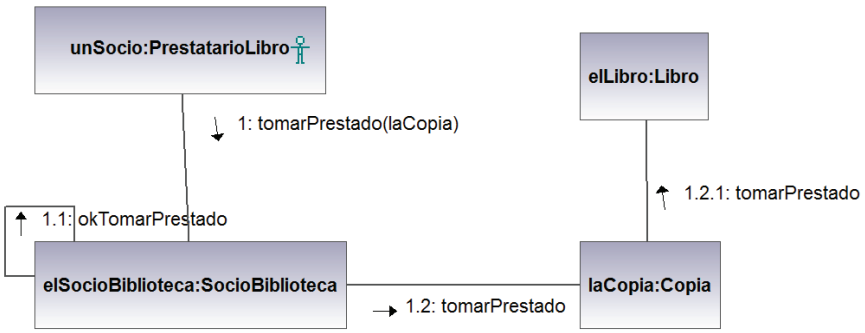


Gráfico A.18 Ejemplo ilustrativo de un diagrama de colaboración correspondiente al diagrama de colaboración del gráfico A.18 [6]

[5] Fuente: STEVENS, Perdita; POOLEY, Rob; Utilización de UML en Ingeniería del Software con Objetos y Componentes; Cap. 9 ; fig. 9.3

[6] Fuente: STEVENS, Perdita; POOLEY, Rob; Utilización de UML en Ingeniería del Software con Objetos y Componentes; Cap. 9 ; fig. 9.2

ANEXO B

B. PROVEEDOR DE SERVICIOS DNS PARA EL JAVA NAMING DIRECTORY INTERFACE™ (JNDI) [7]

Los proveedores de servicios DNS permiten a aplicaciones que implementan JNDI acceder a información almacenada en el Internet Domain Name System. El proveedor presenta el espacio de nombres DNS como un árbol de directorios JNDI y registros de recursos DNS como atributos JNDI.

Cada búsqueda es realizada inicialmente utilizando UDP. Si la respuesta es demasiado larga para ser retornada en un paquete UDP sin fragmentar, la búsqueda se repite a través de TCP.

Un servidor DNS almacena varios tipos de registros de los cuales, para la solución implementada en el proyecto de titulación nos interesan dos de ellos:

Registro A. Del inglés Address (Dirección), usado para la traducción de nombres a su correspondiente dirección IPv4.

Registro PTR. Del inglés Name Pointer (Puntero de Nombre), conocido como *registro inverso*. Realiza la operación inversa al registro A, retornando nombres de dominio.

B.1. Propiedades de Entorno

Las siguientes propiedades el entorno de JNDI son de interés para un proveedor de DNS para la resolución de nombres de dominio. Las propiedades de entorno indican las preferencias que serán usadas por la aplicación para indicar la forma de acceder a los servicios ofrecidos por el proveedor.

java.naming.authoritative

Esta propiedad se utiliza para especificar si todas las respuestas deben ser autorizadas. Si su valor es "true", sólo se aceptan respuestas autorizadas de los servidores DNS, de lo contrario, todas las respuestas son aceptadas. Si esta propiedad no se ha establecido, el valor por defecto es "falso". He aquí un ejemplo que especifica que todas las respuestas deben ser autorizadas.

```
env.put(Context.AUTHORITATIVE, "true");
```

[7] Referencia: Adaptado y traducido de http://java.sun.com/j2se/1.4.2/docs/guide/jndi/spec/

Tenga en cuenta que algunos datos podrían dejar de estar disponibles cuando se solicita que se devuelvan respuestas autorizadas, porque el protocolo DNS no proporciona una forma de solicitar información autorizada. Por ejemplo, el proveedor de servicio DNS podría haber recuperado los datos no autorizados, como resultado de una consulta y posteriormente se ven obligados a descartarse de él porque sólo los datos autorizados pueden ser devueltos.

java.naming.factory.initial

Esta propiedad se utiliza para seleccionar el proveedor de servicios de DNS como el contexto inicial (initial context)[8]. No es utilizado por el propio proveedor. En él se especifica el nombre de clase de una fábrica de contexto inicial (initial context factory)[9] para el proveedor, y puede ser establecido como en el siguiente ejemplo:

```
env.put(Context.INITIAL_CONTEXT_FACTORY,  "com.sun.jndi.dns.DnsContextFactory");
```

java.naming.provider.url

Esta propiedad especifica el nombre de host y el puerto del servidor DNS utilizado por el contexto inicial DNS.

```
env.put(Context.PROVIDER_URL, "dns://server1.sun.com/java.sun.com");
```

La línea anterior permite que el proveedor utilice el servidor DNS en server1.sun.com, y establece el nombre de dominio del contexto inicial como java.sun.com. Si esta propiedad no está definida, el valor predeterminado es "dns:"

Múltiples servidores DNS puede ser especificado al establecer esta propiedad en una lista separada por espacios de direcciones URL. Cada uno de los servidores se pone en contacto a su vez hasta que uno de ellos responde. Desde que el contexto inicial sólo tiene un nombre de dominio único, si son varias URL entonces cada parte debe contener el mismo dominio. Por ejemplo:

```
env.put(Context.PROVIDER_URL,"dns://server1.sun.com/java.sun.com
dns://server2.sun.com/java.sun.com");
```

---

[8] Un contexto indica el tipo de proveedor de servicio de nombres/directorio que será utilizado y las características de su comportamiento.

[9] Clase que crea una instancia de un contexto que será implementado.

com.sun.jndi.dns.recursion

Esta propiedad se utiliza para especificar que la recursividad no está permitida en las consultas DNS. Si esta propiedad no se ha previsto, o si se ha asignado a "true", la recursividad se permite, de lo contrario, la recursividad no está permitida. He aquí un ejemplo que especifica que la recursividad en las consultas DNS no se permiten.

```
env.put("com.sun.jndi.dns.recursion", "false");
```

com.sun.jndi.dns.timeout.initial
com.sun.jndi.dns.timeout.retries

Estas propiedades se usan para alterar los valores predeterminados relacionados con tiempo de espera que el proveedor de DNS utiliza a la hora de presentar las consultas UDP. El proveedor de DNS sostiene UDP consultas mediante el algoritmo exponencial de backoff que se explica a continuación. El proveedor envía una consulta a un servidor DNS y espera una respuesta que debe cumplir en un tiempo de espera (1 segundo por defecto). Si no recibe respuesta en el plazo de tiempo de espera, consulta el siguiente servidor, y así sucesivamente. Si el proveedor no recibe ninguna respuesta por parte de cualquier servidor, se duplica el tiempo de espera y repite el proceso de presentación de la consulta para cada servidor, hasta un número máximo de reintentos (4 por defecto).

La propiedad "com.sun.jndi.dns.timeout.initial", si está establecida, especifica el número de milisegundos que se utiliza como tiempo de espera inicial. Si esta propiedad no se ha establecido, el tiempo de espera predeterminado inicial es de 1000 milisegundos.

La propiedad "com.sun.jndi.dns.timeout.retries", si está establecida, especifica el número de reintentos a usarse en cada servidor DNS con el algoritmo exponencial de backoff descrito previamente. Si esta propiedad no se ha establecido, el número de reintentos predeterminado es de 4.

En la siguiente línea se asigna la espera inicial en 2000 milisegundos y un máximo de tres reintentos.

```
env.put("com.sun.jndi.dns.timeout.initial", "2000");

env.put("com.sun.jndi.dns.timeout.retries", "3");
```

Ejemplo 1:

En este ejemplo se crea un contexto inicial representando el dominio sun.com, luego se lee las direcciones IP tomadas de los correspondientes registros A de dos hosts de ese dominio.

```
Hashtable env = new Hashtable();
env.put("java.naming.factory.initial", "com.sun.jndi.dns.DnsContextFactory");
env.put("java.naming.provider.url", "dns://server1.sun.com/sun.com");
DirContext ictx = new InitialDirContext(env);
Attributes attrs1 = ictx.getAttributes("host1", new String[] {"A"});
Attributes attrs2 = ictx.getAttributes("host2", new String[] {"A"});
```

Ejemplo 2:

La base de datos inversa de DNS tiene su raíz en el dominio de más alto nivel de Internet el Address and Routing Parameter Area (arpa). IPv4 usa el dominio in-addr.arpa.

Para realizar búsquedas de resolución inversa para IPv4 se sigue el siguiente formato:

La secuencia de bytes de la dirección IP es representa en forma inversa, codificada como números decimales separados por puntos y con el sufijo de dominio .in-addr.arpa.

Suponiendo lo siguiente:

Registro A: news. sun.com apunta a 192.130.31.16.

Registro PTR: 16.31.130.192. in-addr.arpa apunta a news. sun.com

Usando el ejemplo anterior para obtener el nombre del dominio.

```
Attributes attrs3 = ictx.getAttributes("16.31.130.192. in-addr.arpa", new String[] {"PTR"});
```

ANEXO C

C. DETALLES DE INSTALACIÓN Y EJECUCIÓN

Las aplicaciones desarrolladas necesitan cumplir con los siguientes requisitos:

1. Sistema Operativo:

   Windows XP o superior.

   Distribución de Linux. (Probada en CentOS 5.2 y Ubuntu 8.04)

2. Tarjeta de Red Fast Ethernet.

3. WinPcap 4.0.2 o superior para Win32.

4. Libpcap 0.8 o superior para Linux.

5. Java Runtime Environment 1.6.0_05 o superior.

6. Creación de la variable de entorno de sistema:

   Windows:

   Copiar el archivo jnetpcap.dll al directorio %SystemRoot%\system32\jnetpcap.dll

   JNETPCAP_HOME con el valor %SystemRoot%\system32\jnetpcap.dll

   Linux:

   Los archivos del paquete jNetPcap de su distribución para Linux (jnetpcap-1.2.rc5-fc8.i386.rpm y jnetpcap-1.2.rc5-deb.i386.deb) se instala de manera predeterminada en

      /usr/lib/libjnetpcap.so

      /usr/share/java/jnetpcap-1.2.rc5.jar

      /usr/share/doc/jnetpcap-1.2.rc5

   Para indicar explícitamente a la aplicación donde encontrar libjnetpcap.so. En CentOS 5.2 debe crearse un archivo en /etc/profile.d [10] que puede llamarse jnetpcap.sh y editarlo de la siguiente manera

   export JNETPCAP_HOME=/usr/lib/libjnetpcap.so

[10] http://planet.admon.org/howto/set-syste-variables-in-debian-and-centos/

El sufijo “.sh” significa que será reconocido como archivo de variables por bash.

En Ubuntu 8.02 debe ejecutar “sudo gedit /etc/environment” y agregar al inicio de este archivo lo siguiente

JNETPCAP_HOME="/usr/lib/libjnetpcap.so"

LD_LIBRARY_PATH="/usr/lib"

La forma de crear variables depende de cada distribución de Linux y debe buscarse documentación sobre este particular en caso de que el usuario lo necesite.

7. Para ejecutar las aplicaciones se debe seguir los siguientes pasos:

- Ubicar los archivo JAR de las aplicaciones incluida la carpeta de bibliotecas en un directorio dedicado para este fin.

- Iniciar una consola y ejecutar la siguiente sentencia:

Para iniciar la captura de paquetes, gráficos en tiempo real y el sniffer.

```
java -Xms32m -Xmx512m -jar “Ruta completa del directorio del programa\TrafficStatistics.jar”
```

Para iniciar la aplicación de generación de resúmenes de datos y gráficos estadísticos.

```
java -Xms32m -Xmx512m -jar “Ruta completa del directorio del programa\ QueryStatistics.jar”
```

Las opciones de la máquina virtual -Xms32m -Xmx512m permiten ampliar los límites de utilización de memoria de la aplicación.

-Xms32m: tamaño inicial de 32MB del espacio de memoria de la JVM.

-Xmx512m: tamaño máximo de 512MB del espacio de memoria de la JVM.

Si se sobrepasa este valor la aplicación en ejecución lanzará la excepción “`java.lang.OutOfMemoryError: Java heap space`”. Es recomendable no asignar el máximo de memoria de la estación de trabajo en este parámetro para evitar conflictos con otras aplicaciones o el sistema operativo.

Estas opciones se incluyen como precaución en caso de que la aplicación tenga que almacenar en memoria grandes cantidades de datos. Por ejemplo en el caso de la captura de paquetes, los datos de tráfico son almacenados en listas dinámicas previa a la actualización de la base de datos y posterior

liberación de memoria, esto se vuelve crítico si se monitorea una red con una conexión a Internet de alta tasa de transferencia y muchas estaciones de trabajo activas. Otro ejemplo importante es la generación de gráficos y resúmenes de tráfico de Internet; pues previa a la visualización de los resultados debe mantenerse en memoria los valores de la consulta a la base de datos, más aún si es de un gran lapso de tiempo.

-La base de datos se crea de manera predeterminada en los siguientes directorios:

Windows:

“Ruta completa del directorio de Java\DATABASE”

Linux:

“/home/DATABASE”

- El software puede ser instalado en los ambientes de red ilustrados por los gráficos que se muestran a continuación:

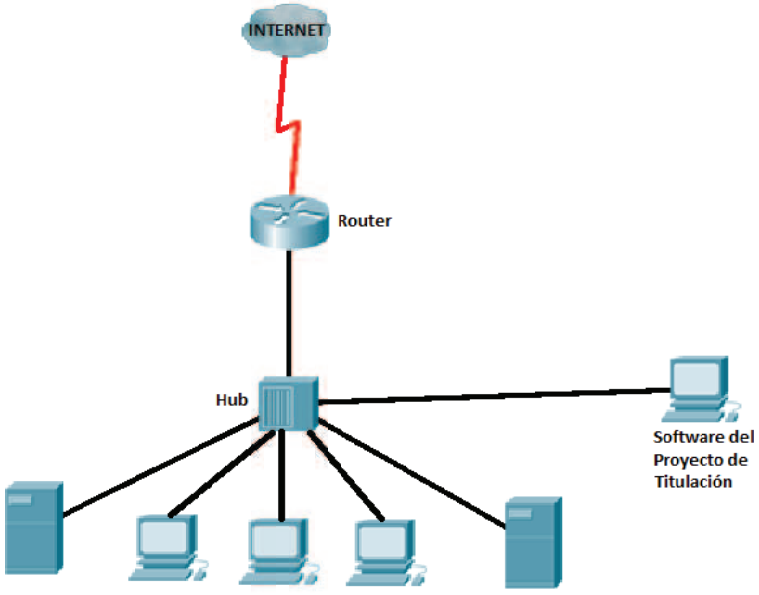


Gráfico C.1 Red compartida mediante el uso de un Hub

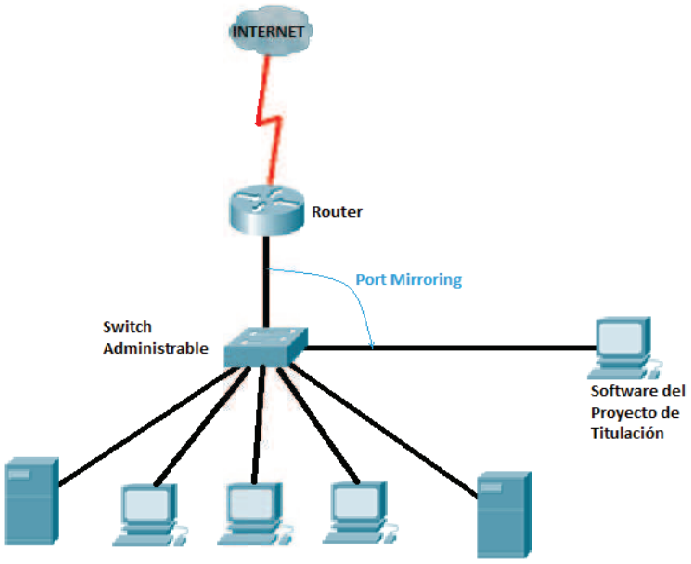


Gráfico C.2 Red conmutada mediante un Switch Administrable

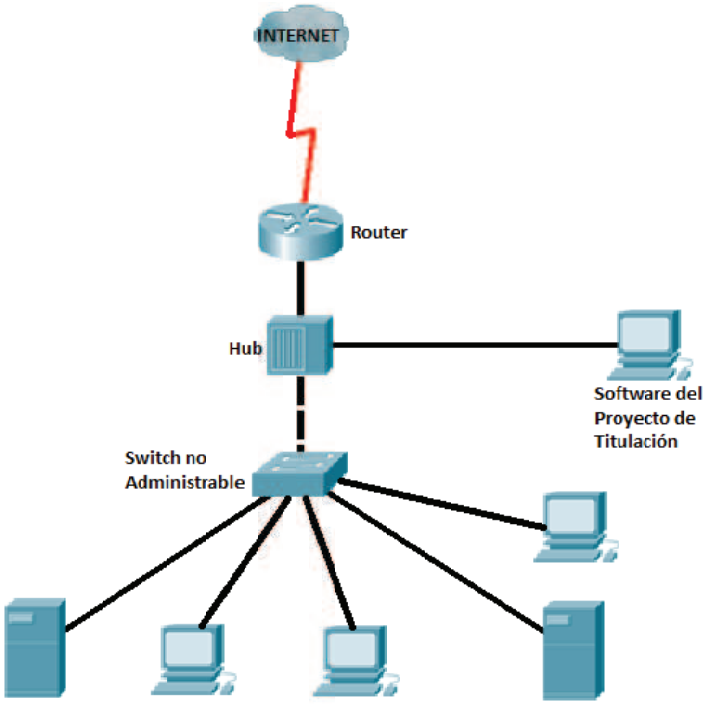


Gráfico C.3 Red conmutada mediante un Switch no Administrable

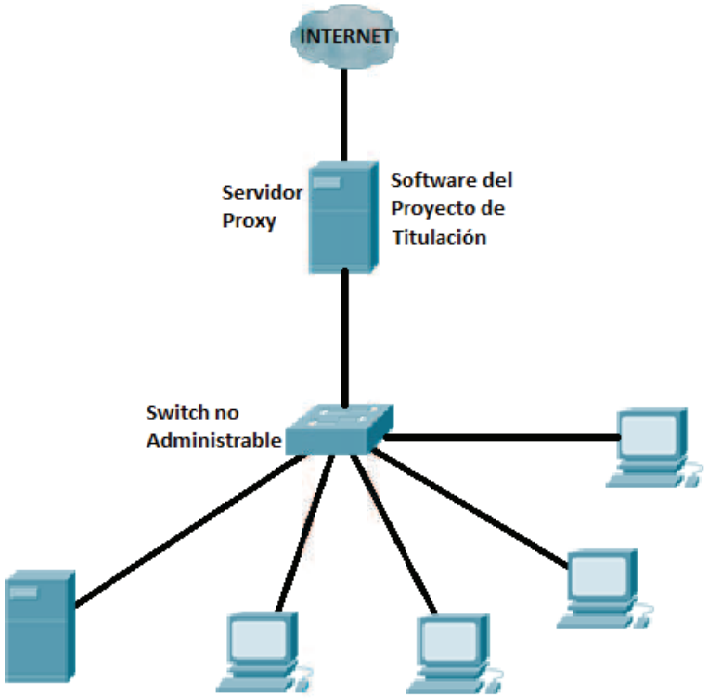


Gráfico C.4 Red conmutada mediante un Switch no Administrable y que usa un servidor Proxy

ANEXO D

D. COMPARACIÓN DETALLADA DEL SOFTWARE DESARROLLADO CON WIRESHARK WIN32-1.2.8 Y COLASOFT CAPSA 7.1

Para poder comparar los tres tipos de software de una manera más clara de ha divido el análisis según los siguientes criterios:

- Análisis y captura de datos en tiempo real.
- Almacenamiento de datos.
- Obtención y análisis de datos almacenados.

D.1. Análisis y Captura de datos en tiempo Real

D.1.1. Selección de la interfaz de red.

D.1.1.1. Proyecto de titulación

Para la selección de la interfaz de red se ha provisto de un botón con una imagen descriptiva en la barra principal de herramientas, que permite elegir el dispositivo deseado por el usuario.

Gráfico D.1 Botón de selección de tarjeta de Red (Proyecto).

Posteriormente aparecerá una ventana de diálogo con las diferentes interfaces de red que posee el computador, de esta manera el usuario elegirá el dispositivo para el monitoreo del tráfico de Internet.

Gráfico D.2 Selección Dispositivo de Red (Proyecto).

D.1.1.2. Wireshark win32-1.2.8

Para elegir la interfaz de red, Wireshark provee un botón en la barra principal de herramientas.

Gráfico D.3 Botón de selección de tarjeta de Red (Wireshark).

Al pulsar el botón de elección de interfaz de red, aparece un cuadro de diálogo con los dispositivos de red que posee la máquina en donde se encuentra instalado el software para el monitoreo.

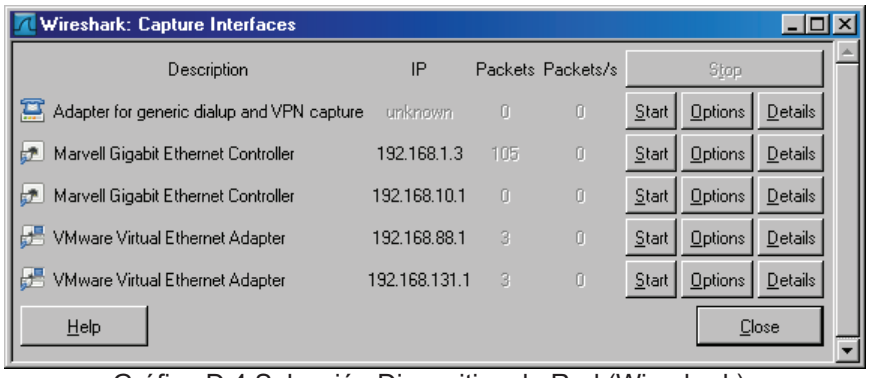


Gráfico D.4 Selección Dispositivo de Red (Wireshark).

Además las opciones mostradas en el gráfico D.4 muestran un cuadro de diálogo que permite configurar si la tarjeta de red es local o remota, así como otras opciones de filtros y detalles de la tarjeta de red que más adelante se describirán con más profundidad.

D.1.1.3. Colasoft Capsa 7.1

Este software a diferencia de los anteriores posee una ventana de inicio que permite elegir la interfaz de red de una lista y muestra la actividad de la tarjeta señalada. Esta elección de dispositivo de red se la realiza antes de iniciar la captura.

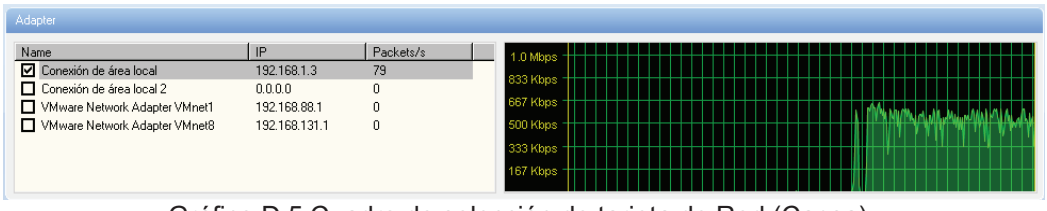


Gráfico D.5 Cuadro de selección de tarjeta de Red (Capsa).

Además contiene una barra de herramientas que permite a través de un botón, cambiar la interfaz de red, desplegando el cuadro de selección anterior.

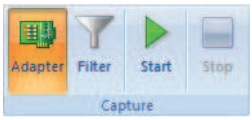


Gráfico D.6 Botón de selección de tarjeta de Red (Capsa).

D.1.2. Selección de estaciones de trabajo.

D.1.2.1. Proyecto de titulación

Como primer paso se debe iniciar la captura de datos para poder así habilitar las funciones de selección de hosts monitoreados y filtros, útiles en la representación de los diferentes valores de cada paquete involucrado en la transferencia de información.

Gráfico D.7 Botón de inicio de Captura (Proyecto).

Después de haber iniciado la captura global de datos se mostrará en la parte izquierda de la pantalla un panel que contiene un árbol de estaciones de trabajo obtenidas del monitoreo actual y que son descritas por su dirección IP, al extender este nodo aparece una casilla de selección para la dirección IP actual y su correspondiente MAC.

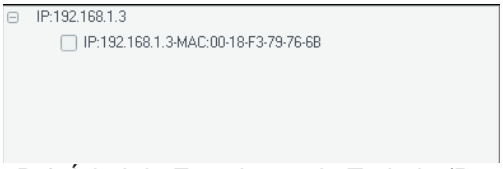


Gráfico D.8 Árbol de Estaciones de Trabajo (Proyecto).

D.1.2.2. Wireshark win32-1.2.8

Contiene un campo de texto (Filter) que permite implementar un filtro, el cual tiene la capacidad de asignar una dirección IP o un rango de estas, dependiendo del análisis que requiera el usuario para aplicar a los paquetes que están siendo capturados.

Gráfico D.9 Filtro para selección de estaciones de trabajo (Wireshark).

D.1.2.3. Colasoft Capsa 7.1

En la ventana de inicio, el programa implementa un cuadro de diálogo que permite añadir estaciones de trabajo para realizar el monitoreo de datos de los hosts que el usuario especifique para el análisis.

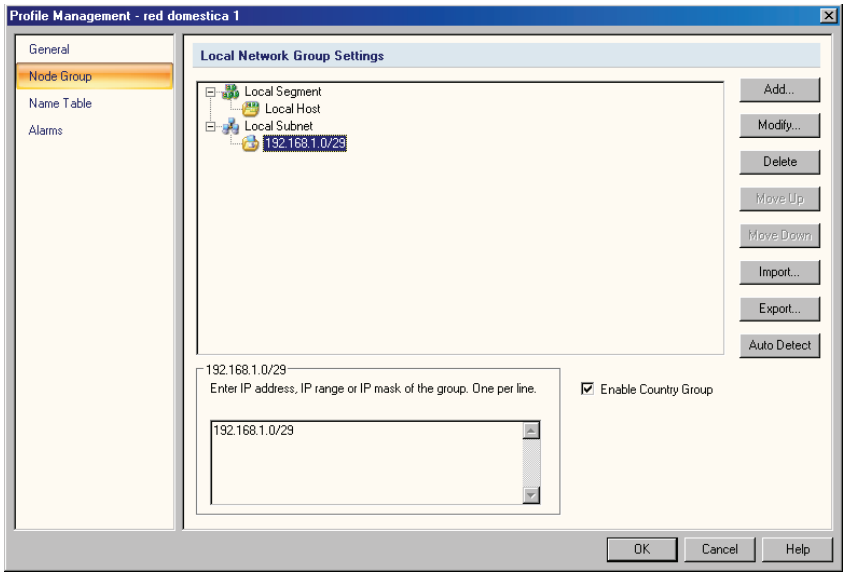


Gráfico D.10 Cuadro para añadir estaciones de trabajo (Capsa).

En caso de no asignar estaciones de trabajo en el cuadro de diálogo anterior, el usuario puede elegir de los hosts monitoreados dentro de un panel situado en la parte izquierda de la pantalla mientras se realiza el monitoreo del tráfico de Internet.

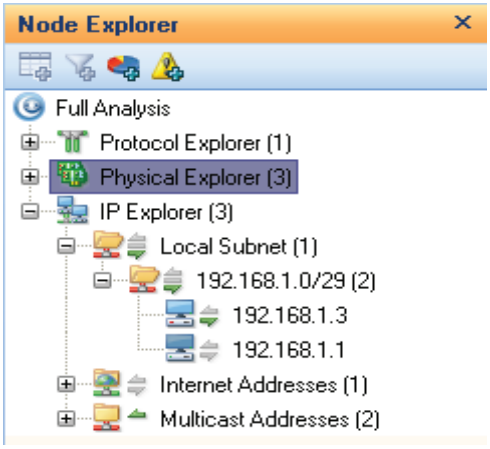


Gráfico D.11 Panel para selección de estaciones de trabajo (Capsa).

D.1.3. Filtros y Análisis de paquetes.

D.1.3.1. Proyecto de titulación

El filtro implementado en el proyecto de titulación consiste en una pestaña llamada Modo Sniffer, la cual contiene una barra de herramientas con un combo que permite la selección de un protocolo TCP, datagrama UDP, un mensaje ICMP o todos ALL, con lo cual los paquetes involucrados en la transferencia de información se imprimirán en un área de texto en un formato simplificado.

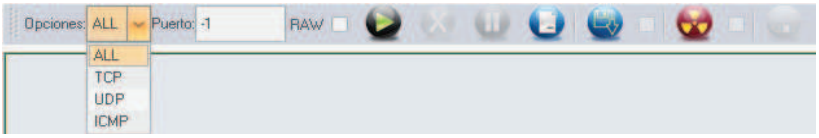


Gráfico D.12 Combo de Selección de Protocolo (Proyecto).

Además posee un cuadro de texto el cual representa el número de puerto del filtro a asignar para la impresión de los paquetes monitoreados.

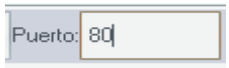


Gráfico D.13 Cuadro de texto selección de puerto (Proyecto).

Se ha añadido una opción representada por una casilla RAW, la cual permite imprimir el contenido completo de los paquetes monitoreados en un formato legible, de acuerdo a la elección de filtros y las estaciones de trabajo seleccionadas en el árbol.

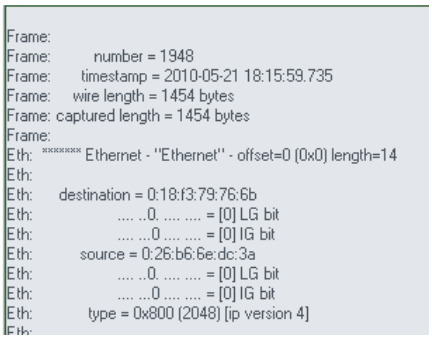


Gráfico D.14 Área de texto de impresión de paquetes monitoreados (Proyecto).

D.1.3.2. Wireshark win32-1.2.8

Posee una barra de herramientas, la cual contiene un campo de texto donde se especificará el filtro que se desea aplicar a los paquetes que están siendo capturados.

Gráfico D.15 Filtro para selección de protocolo y puerto (Wireshark).

Además contiene tres botones que permiten mostrar un cuadro de diálogo para asignar el tipo de filtro que se va a configurar para el tráfico monitoreado.

Gráfico D.16 Botón Filtro de Expresiones y funciones lógicas (Wireshark).

En el cuadro de diálogo despliega todos los tipos de filtros que Wireshark contiene, además permite hacer uso de funciones lógicas para un mejor filtrado.

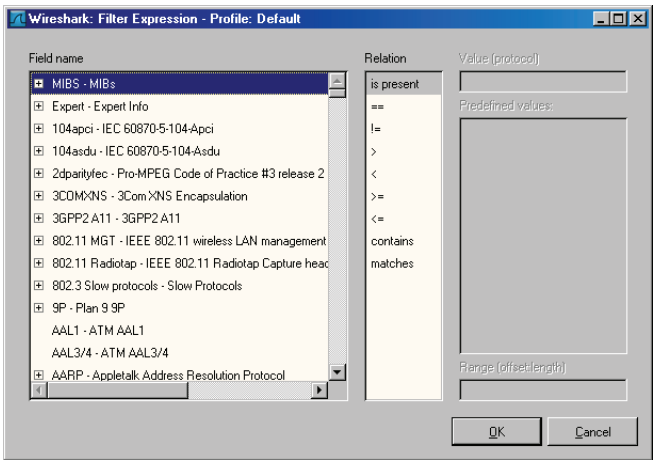


Gráfico D.17 Cuadro Filtro de Expresiones y funciones lógicas (Wireshark).

D.1.3.3. Colasoft Capsa 7.1

Este software posee ventana de inicio la cual permite configurar los diferentes tipos de filtros a través un cuadro de diálogo y un panel gráfico.

Los filtros elegidos por el usuario se pueden exportar e importar para facilitar el monitoreo.

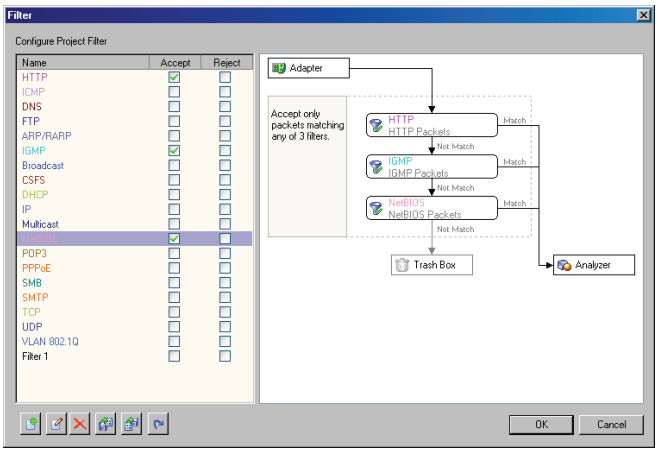


Gráfico D.18 Cuadro para añadir filtros en cascada (Capsa).

Además posee una barra de herramientas con un botón Filter que permite asignar un nuevo filtro dentro de la captura de datos.

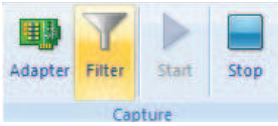


Gráfico D.19 Botón para selección de Filtro (Capsa).

D.1.4. Gráficos en Tiempo Real

D.1.4.1. Proyecto de titulación

Gráficos en tiempo real totales.

Se ha implementado varios gráficos en tiempo real, dos de ellos representan el tráfico total de la red, uno corresponde al tráfico incoming (entrante) y el otro al outgoing (saliente). Estos gráficos tienen la propiedad de asignar un rango de colores según el límite máximo de transferencia de datos asignado por el usuario.

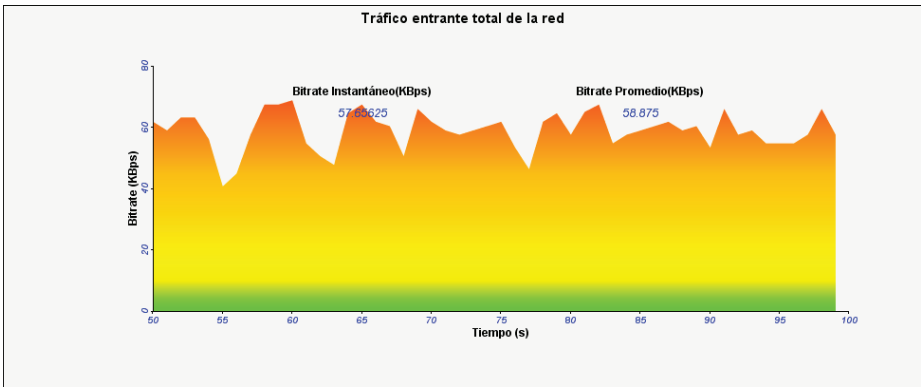


Gráfico D.20 Bitrate vs Tiempo incoming (Proyecto).

El gráfico D.20 representa el tráfico incoming mostrando detalles sobre el bitrate instantáneo y el bitrate promedio. El rango de colores tiene una asignación especial para los valores máximos que está representado por el color rojo, para valores mínimos el color verde, para los valores intermedios van variando y cambiando de color conforme vayan acercándose al valor máximo asignado por el usuario. En este ejemplo se ha asignado el valor límite en 100 KBps por ello toma un color de transición de verde a naranja.

El gráfico D.21 representa el tráfico outgoing y se ha asignado un valor máximo de transferencia de datos en 25 KBps, de la misma manera muestra los valores de bitrate instantáneo y bitrate promedio con un rango de colores dependiendo de los valores máximos asignados por el usuario, razón por la cual el gráfico adopta los siguientes colores para la representación de la transferencia de datos.

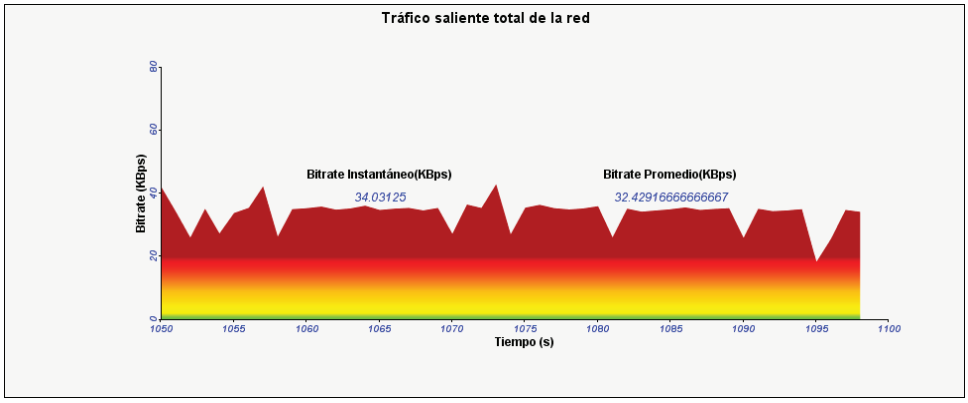


Gráfico D.21 Bitrate vs Tiempo outgoing con valor límite 25KBps (Proyecto).

A diferencia de los demás software este hace una clara distinción entre tráfico incoming y outgoing separando los gráficos para una mejor interpretación de los datos.

Para poder respaldar este tipo de información se ha añadido botones en la barra de representación gráfica que permiten guardar los diagramas en archivos con extensión jpeg, png, bmp.

Por otro lado se agrego un botón para reiniciar la representación gráfica de la transferencia de datos.

Gráfico D.22 Botónes de Reinicio y captura de gráficos en tiempo real (Proyecto).

Una de las herramientas más útiles para un administrador de red es tener las aplicaciones de monitoreo minimizadas en el área de notificación de la barra de tareas, ya que mientras se utilizan otras aplicaciones se puede vigilar a simple vista la utilización del acceso a Internet.

Por ello se ha implementado para los gráficos de bitrate total en tiempo real dos "tray icons"[11] mostrados en la barra de tareas tanto para el tráfico saliente y tráfico entrante.

Gráfico D.23 Tray Icons de gráficos en tiempo real incoming y outgoing (Proyecto).

Gráficos en tiempo real por estación de trabajo.

Al implementar un árbol de hosts obtenido del monitoreo de datos, se añadió la posibilidad de presentar gráficos en tiempo real de los hosts seleccionados, esto permite obtener valiosa información de uno o varios hosts mientras transmitan o reciban datos hacia Internet.

Por esta razón se añadió una pestaña llamada "tráfico en tiempo real por hosts seleccionados", la cual contiene dos gráficos que representan al tráfico de incoming y outgoing de los hosts seleccionados en el árbol de estaciones de trabajo.

Se muestra a continuación el tráfico incoming de los hosts seleccionados con los valores de bitrate instantáneo y bitrate promedio, con un rango de colores dependiendo del valor máximo asignado por el usuario.

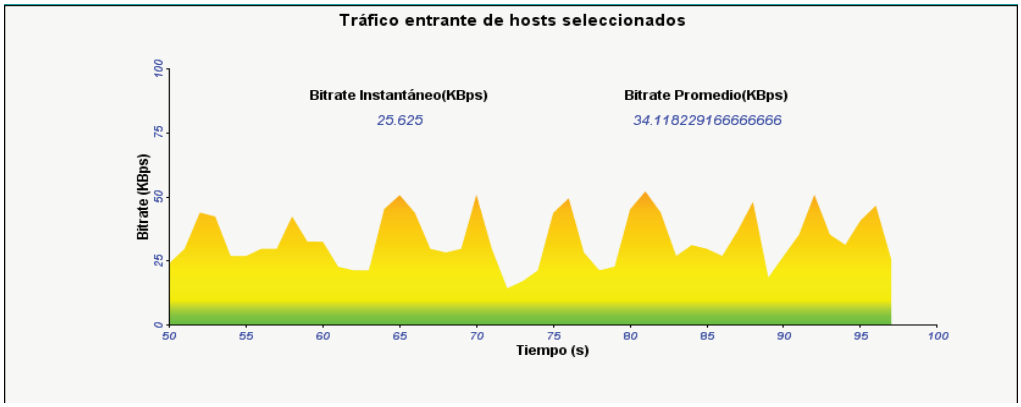


Gráfico D.24 Bitrate vs Tiempo incoming para los hosts seleccionados (Proyecto).

[11] Íconos del área de notificación.

Como se puede apreciar en el gráfico D.25 si se seleccionan todos los hosts del árbol de estaciones de trabajo se obtiene un gráfico similar al del tráfico total en tiempo real.

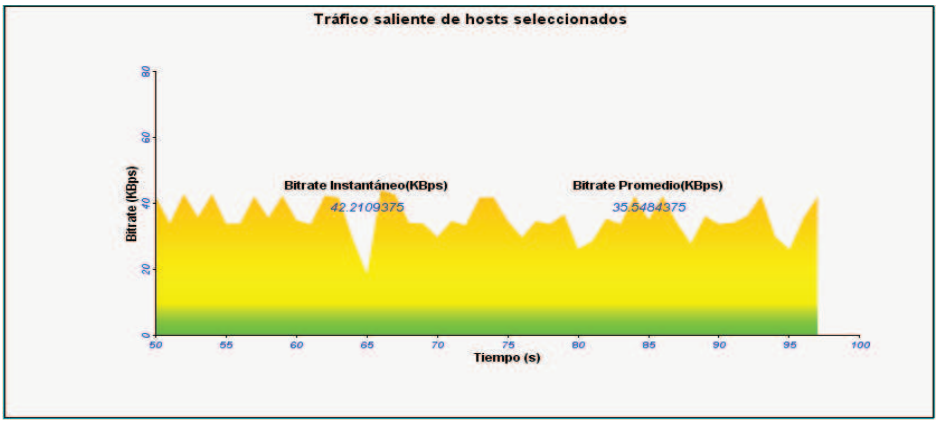


Gráfico D.25 Bitrate vs Tiempo outgoing para los hosts seleccionados (Proyecto).

Al igual que los gráficos de tráfico total se añadió una barra de herramientas que permiten guardar las representaciones gráficas en archivos con extensión jpeg, png, bmp.

También se incluye un botón para reiniciar el gráfico y otro para sincronizar los gráficos de tráfico de hosts seleccionados y de tráfico total de la red en un mismo instante de tiempo.

Gráfico D.26 Botónes para igualar, reiniciar, y capturar gráficos en tiempo real de los hosts seleccionados (Proyecto).

De la misma forma se añadió "tray icons" para los gráficos del tráfico de hosts seleccionados.

Gráfico D.27 Tray Icons de gráficos en tiempo real incoming y outgoing por hosts seleccionados (Proyecto).

D.1.4.2. Wireshark win32-1.2.8

En Wireshark los diagramas de representación de datos en tiempo real dependen de los filtros asignados por el usuario. El gráfico de transferencia total de datos tiene preseleccionado por defecto el color negro, para los demás gráficos se puede seleccionar en el panel los diferentes colores que se pueden

asignar de acuerdo al filtro que se aplique. En este caso se asignó para demostración un filtro para tráfico http y este se muestra con color rojo.

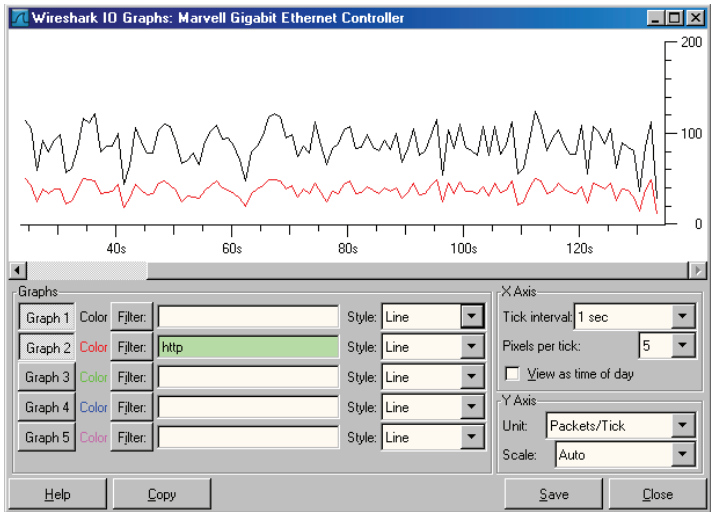


Gráfico D.28 Diagrama de representación gráfica de líneas para los filtros (Wireshark).

Los gráficos en tiempo real se pueden representar en cuatro formas diferentes que son: puntos, líneas, impulsos y barras, los cuales pueden elegirse de manera indistinta.

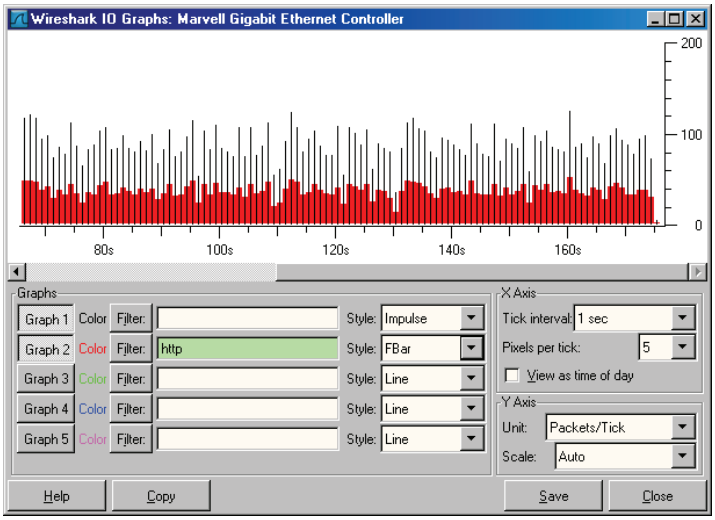


Gráfico D.29 Diagrama de representación gráfica de impulsos para los filtros (Wireshark).

Los gráficos se pueden representar con una combinación de líneas para un tipo de tráfico y puntos para otro, con intervalos de tiempo en el eje horizontal a 1 segundo, 10 o 60 segundos; de igual forma en el eje vertical se puede seleccionar la unidad de paquetes, bits o bytes.

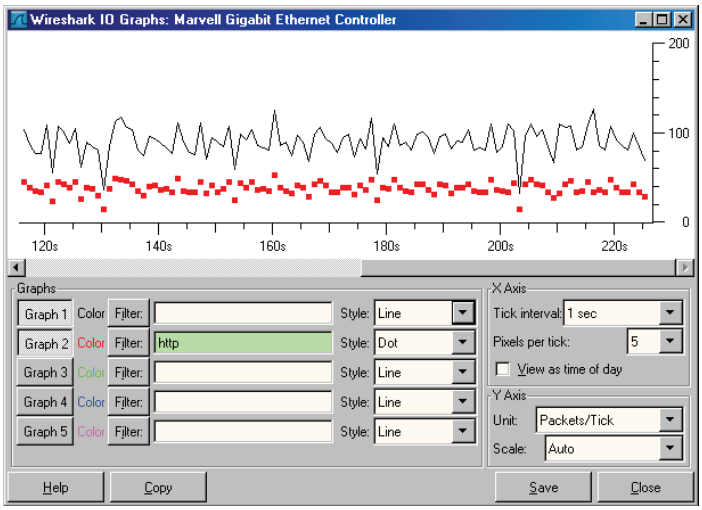


Gráfico D.30 Diagrama de representación gráfica de puntos para los filtros (Wireshark).

Para realizar un análisis de tráfico incoming y outgoing con Wireshark, el usuario deberá asignar tantos filtros crea necesarios para obtener solo el tráfico entrante de toda la transferencia de datos. De idéntica forma para capturar solo el tráfico saliente.

Esto no representa una solución real para un usuario que no esté complemente familiarizado con la aplicación.

Además si se requiere una representación gráfica del tráfico incoming y outgoing de varios hosts seleccionados el usuario deberá implementar filtros y operadores lógicos para poder filtrar los datos de mejor manera, lo que resulta tedioso para un usuario no experimentado.

Los filtros de Wireshark son muy restrictivos, es decir, en el caso de que se desee monitorear otra estación de trabajo y observar la gráfica del tráfico en tiempo real, es necesario agregarlo manualmente, ya que no es capaz de detectarlo automáticamente.

Por último Wireshark no contiene “tray icons” de ningún tipo en tiempo real, esto dificulta la realización de otras actividades concernientes a la administración de la red sin que se descuide el monitoreo del ancho de banda.

D.1.4.3. Colasoft Capsa 7.1

Este software contiene varios gráficos en tiempo real como un medidor de la tasa de transferencia de los datos, un medidor de paquetes por segundo, un gráfico del historial de la transferencia de paquetes y un gráfico de porcentaje de memoria utilizado para almacenar los paquetes capturados.

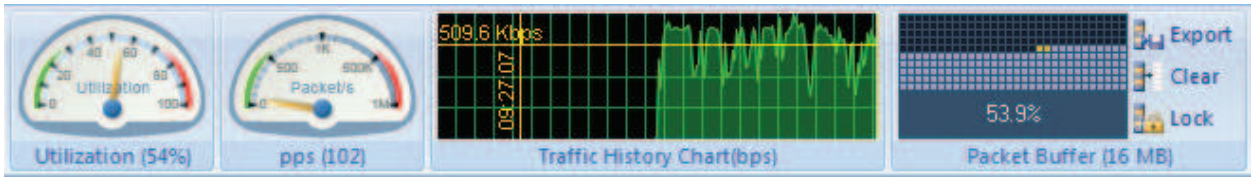


Gráfico D.31 Representaciones gráficas en tiempo real (Capsa).

Contiene un diagrama de bytes por segundo en tiempo real de la transferencia de datos total.

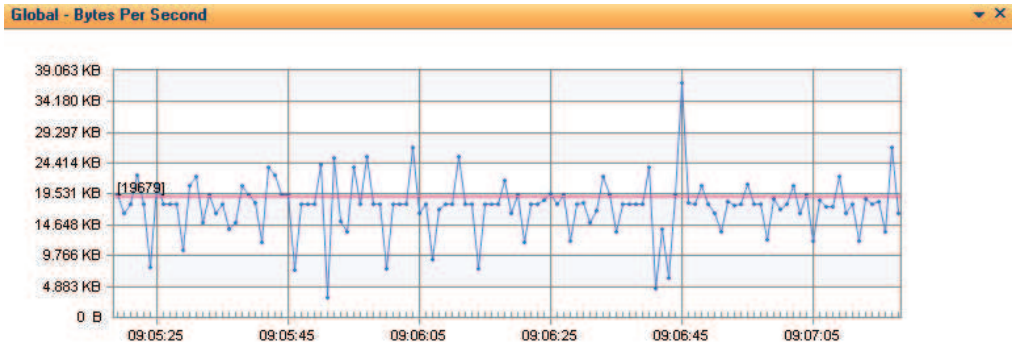


Gráfico D.32 Bytes por segundo en tiempo real (Capsa).

Contiene una representación gráfica de barras en tiempo real correspondientes a los valores de bytes por segundo de la transferencia de datos total.

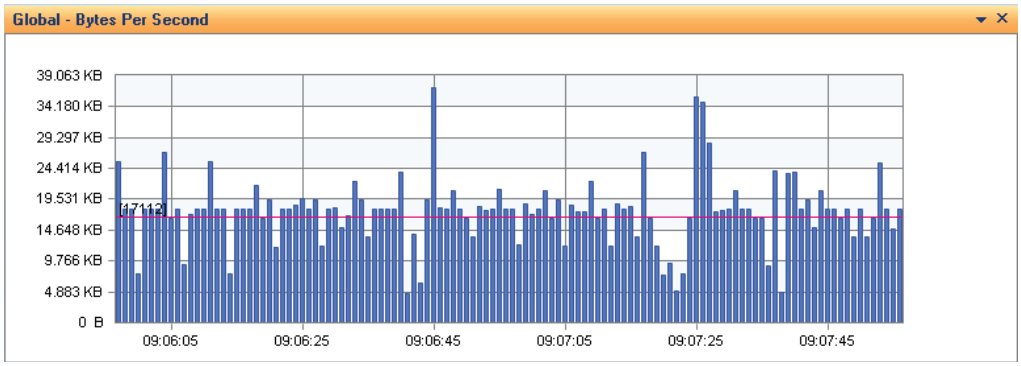


Gráfico D.33 Bytes por segundo con representación de barras en tiempo real (Capsa).

Puede crear gráficos en tiempo real para las siguientes opciones del software llamando a un cuadro de diálogo para su selección.

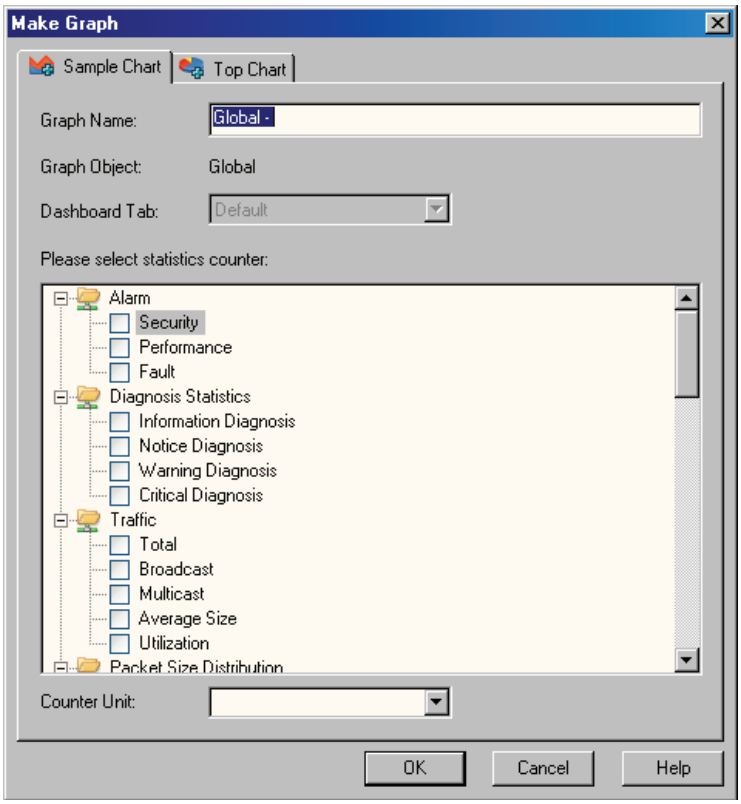


Gráfico D.34 Cuadro para selección de gráficos en tiempo real (Capsa).

Esta clasificación de gráficos se localiza en la pestaña “sample chart”, que se los lista a continuación:

- Alarma: seguridad, rendimiento y fallos.
- Tráfico: Total, Broadcast, Multicast Drop, tamaño promedio y Utilización.
- Distribución de paquetes (Tamaño): <= 64, 65-127, 128-255, 246-511, 512-1023, 1024-1517 y> = 1518.
- Dirección: direcciones físicas, direcciones IP, dirección IP local y dirección IP remota.
- Protocolo: protocolos de enlace de datos, protocolos de red, protocolos de transporte, protocolos de sesión, protocolos capa presentación, protocolo de capa aplicación.
- Descarga: Conversación física, conversación IP, la conversación TCP y UDP.
- TCP: banderas SYN TCP enviados, SYNACK TCP enviados, FIN TCP enviados y TCP Reset enviados.

- Análisis de DNS: DNS de consultas y la respuesta de DNS.
- Análisis de correo electrónico: La conexión SMTP y POP3 de conexión.
- Análisis de FTP: Subidas FTP y descargas FTP.
- Análisis de HTTP: Petición HTTP, conexión HTTP y Servidor HTTP

A continuación se muestra los diagramas más relevantes de la lista anterior.

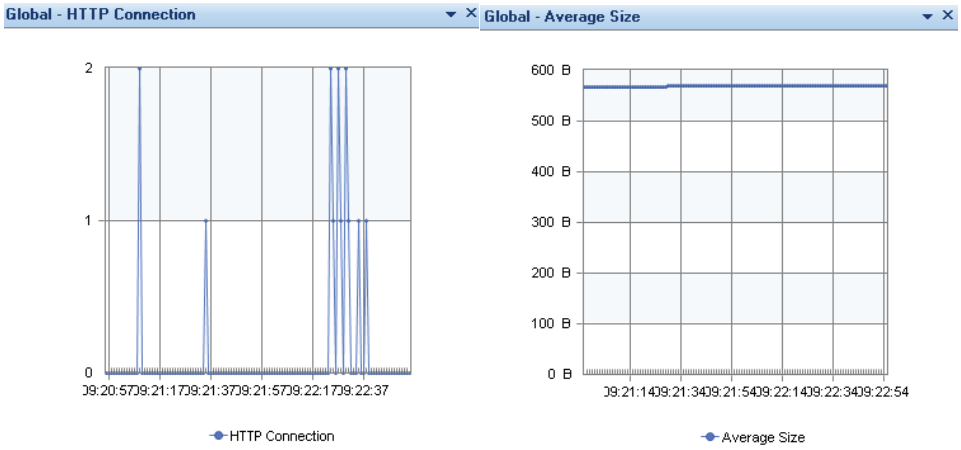


Gráfico D.35 Conexión Http y promedio en tiempo real (Capsa).

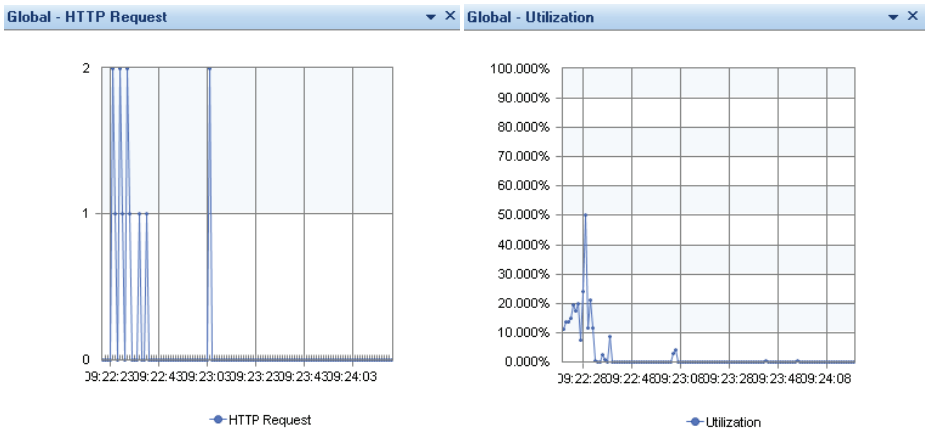


Gráfico D.36 Petición Http y utilización en tiempo real (Capsa).

Hay que tener en cuenta que para la representación de este tipo de gráficos hay que asignar grupos de hosts, ya que son gráficos globales más no gráficos para representaciones unitarias.

Si el usuario requeriera un tipo de gráfico unitario o simplemente de pocas máquinas requerirá añadir un grupo o un host para monitorear la transferencia de información.

Este tipo de gráficos en tiempo real no hace una distinción entre tráfico entrante y saliente, por lo cual si un grupo de hosts están subiendo datos y otro grupo de hosts están descargando datos, el programa suma toda esa cantidad de transferencia como si fuera una sola, sin ninguna distinción.

Para muchas de las empresas los dos tipos de ancho de banda de subida y bajada son parámetros fundamentales para el análisis de una red.

El software permite añadir otros tipos de gráficos, y se los lista en el idioma inglés ya que el software no posee multilenguaje y se quiere mantener intacto el significado original.

Esta clasificación de gráficos se localiza en la pestaña “top chart”.

- Top Physical Group Total Traffic.
- Top Physical Group Traffic Received.
- Top Physical Group Traffic Sent.
- Top IP Group Total Traffic.
- Top IP Group Traffic Received.
- Top IP Group Traffic Sent.
- Top Physical Address Total Traffic.
- Top Physical Address Traffic Received.
- Top Physical Address Traffic Sent.
- Top IP Address Total Traffic.
- Top Local IP Address Total Traffic.
- Top Remote IP Address Total Traffic.
- Top IP Address Traffic Received.
- Top IP Address Traffic Sent.
- Top Local IP Address Traffic Received.

- Top Local IP Address Traffic Sent.
- Top Remote IP Address Traffic Received.
- Top Remote IP Address Traffic Sent.
- Top Application Protocols.
- Packet Size Distribution.

De la lista anterior de gráficos, solo se mostrarán los más relevantes.

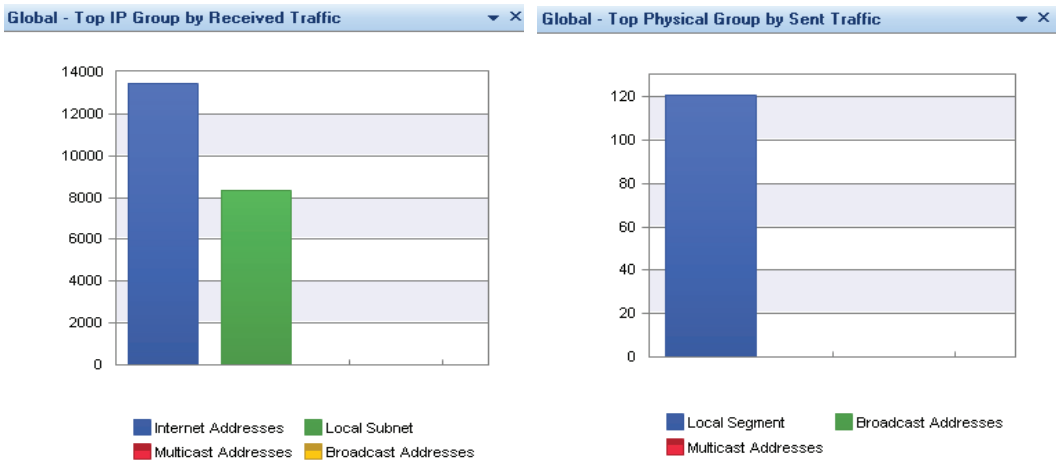


Gráfico D.37 Tráfico Entrante y tráfico Saliente en tiempo real (Capsa).

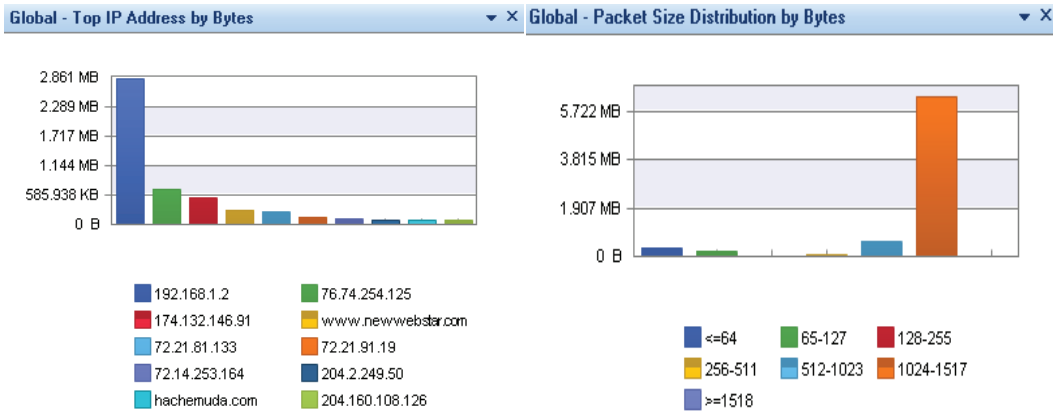


Gráfico D.38 Direcciones IP y tamaño de paquetes en tiempo real (Capsa).

Los gráficos anteriores necesitan de un buffer asignado por el usuario para poder realizar sus diagramas, la actualización y precisión de los mismos depende de ello, por esta razón este tipo de gráficos no se deberían considerar en tiempo real puro, sino en un tiempo real con retardo de actualización.

D.1.5. Alarmas

D.1.5.1. Proyecto de titulación

Para el proyecto de titulación se añadió una alarma, la cual le permite al usuario o administrador de red editar un valor máximo de tasa de transferencia de datos a través de Internet. La alarma se puede asignar para el tráfico total de la red.

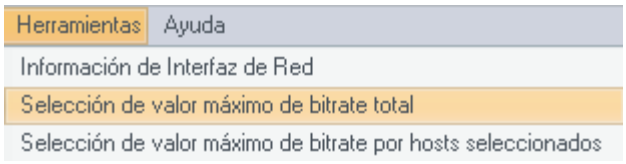


Gráfico D.39 Alarma para tráfico total (Proyecto).

De igual manera la alarma se puede asignar para el tráfico de hosts seleccionados.

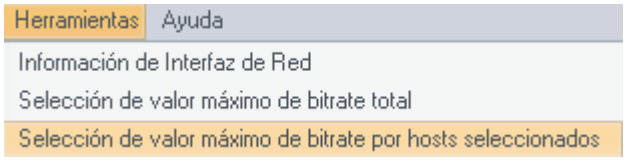


Gráfico D.40 Alarma para tráfico por hosts seleccionado (Proyecto).

Para asignar el valor máximo de alarma se cambia el valor por defecto de bitrate del cuadro de diálogo, si se ingresa valores negativos automáticamente se restablece el valor por defecto.

No se ha considerado alarmas por debajo de 10 KBps ya que el programa está orientado a monitorear tráfico de Internet de varios hosts.

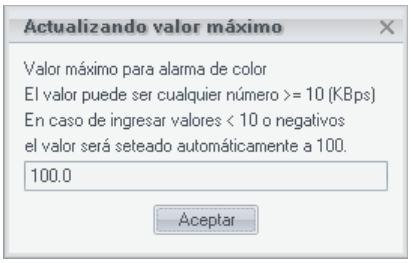


Gráfico D.41 Cuadro para asignar el valor máximo de alarma (Proyecto).

Como consecuencia de haber asignado una alarma para los gráficos totales de red y para los gráficos por hosts seleccionados, se produce una alarma sonora (beep) cuando se sobrepasa este límite.

De la misma manera se visualiza que el color del gráfico D.42 se vuelve rojo si los valores de bitrate superan el límite de alarma asignado por el usuario.

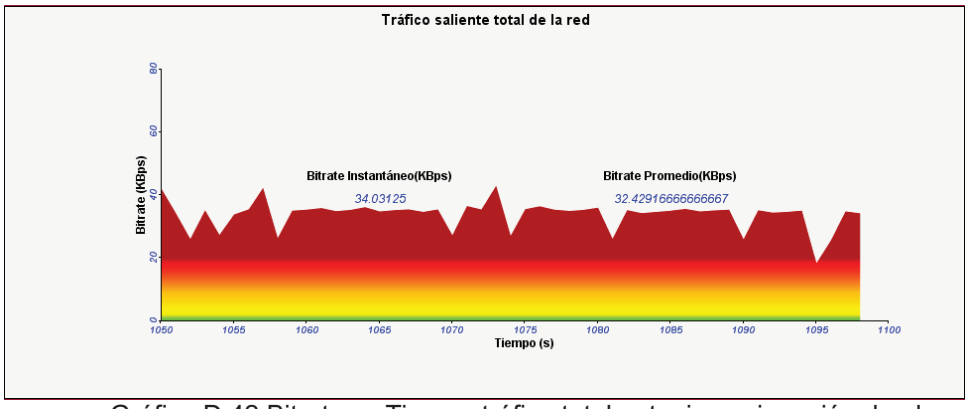


Gráfico D.42 Bitrate vs Tiempo tráfico total outgoing asignación de alarma a 20 KBps (Proyecto).

D.1.5.2. Wireshark win32-1.2.8

Este de software de análisis de paquetes no posee ninguna función de alarma para el monitoreo de tráfico.

D.1.5.3. Colasoft Capsa 7.1

Este software ha implementado las alarmas en la ventana de inicio, de esta forma el usuario puede asignarlas tanto para seguridad y rendimiento del tráfico total, a través de un cuadro de diálogo.

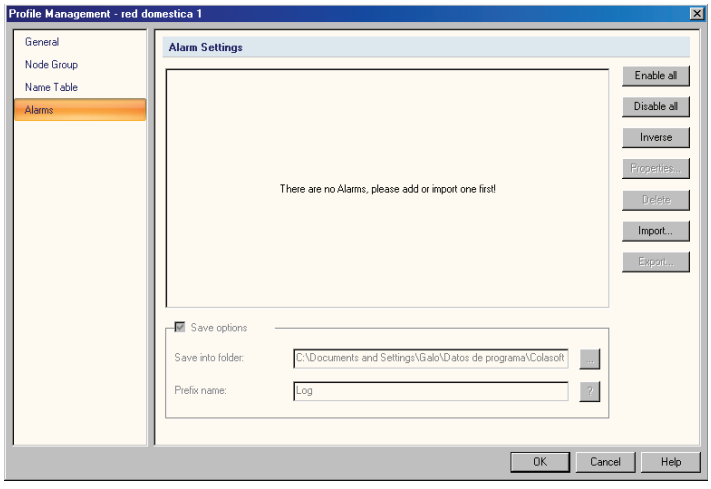


Gráfico D.43 Cuadro para añadir alarmas (Capsa).

Una vez iniciada la captura de datos, se puede asignar alarmas con el botón “Alarm Settings” en la barra de tareas.

Gráfico D.44 Botón asignación de alarmas (Capsa).

Otra forma de asignar alarmas en el cuadro “Node Explorer”.

Gráfico D.45 Botón asignación de alarmas en el cuadro Node (Capsa).

Una vez seleccionado el botón “asignar alarma” se muestra el siguiente cuadro de diálogo.

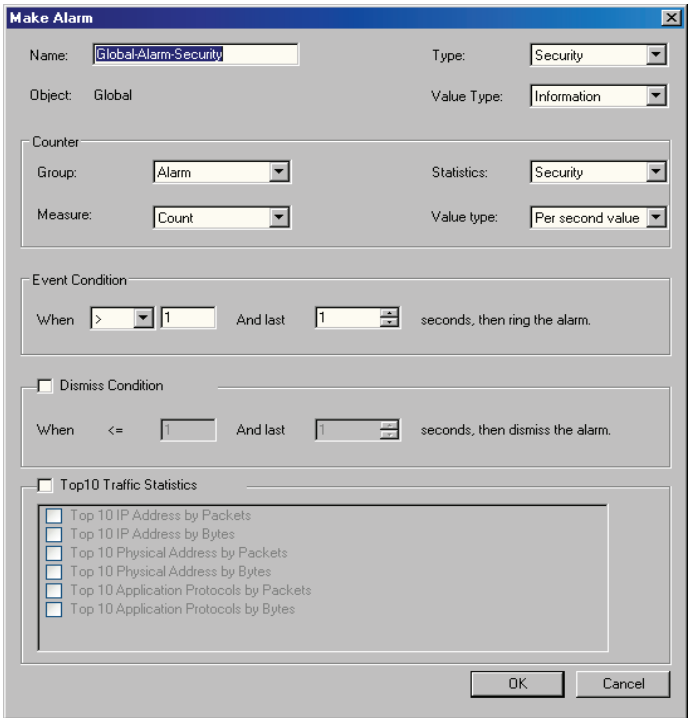


Gráfico D.46 Cuadro de diálogo para asignación de alarma cuadro Node (Capsa).

De esta manera se puede crear un tipo de alarma para seguridad o rendimiento.

Cuando la alarma que se ha asignado en el monitoreo de tráfico de Internet se activa se muestra el siguiente cuadro.

Gráfico D.47 Cuadro de alarma Node (Capsa).

D.1.6. Almacenamiento de Datos

D.1.6.1. Proyecto de titulación.

En el proyecto de titulación existen dos formas para respaldar los datos obtenidos en el monitoreo del tráfico de Internet.

La primera forma esta asignada en la pestaña de Modo Sniffer, aquí se respalda la información de los paquetes monitoreados en archivos de texto, ya sea en varios archivos o en un solo archivo de texto según requiera el usuario.

Estos archivos contendrán la información de las cabeceras de los paquetes ya sea en modo RAW (paquete completo decodificado) o modo NO RAW(información corta de cabeceras de protocolo decodificadas).

Gráfico D.48 Botones para limpiar área de texto, guardar automáticamente y no guardar datos monitoreados (Proyecto).

Los botones del gráfico D.48 descritos en el diseño de la interfaz de usuario del capítulo 3 están configurados así, ya que si no se ha asignado ningún método de guardar los datos automáticamente, despliega un cuadro de diálogo que permite informar al usuario si desea guardar la información.

Si después de 5 segundos no hay respuesta un contador permite cerrar el cuadro de diálogo.

El cuadro de diálogo de despliega cada vez que en el área de texto existan más de 15000 líneas de información.

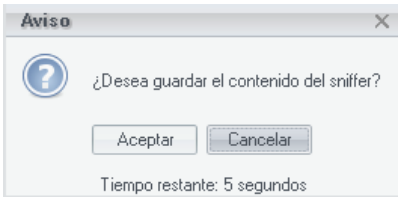


Gráfico D.49 Cuadro de diálogo para respaldar datos monitoreados en el sniffer (Proyecto).

Si el usuario casualmente solo requiere guardar el texto presente en ese momento, existe un cuarto botón que permite almacenar solo esa información.

Gráfico D.50 Botones para limpiar área de texto, guardar datos monitoreados (Proyecto).

Se ha comprobado que esta es la forma en la que Wireshark y Capsa almacenan la información, ya sea en archivos contiguos o en un solo archivo que puede llegar a pesar hasta 10 GB.

La segunda forma de almacenamiento de datos y la más importante es la implementación de una base de datos para respaldar los valores obtenidos durante el monitoreo.

Se añadido dos botones que permiten exportar e importar la base de datos tanto para la aplicación de monitoreo como para la aplicación de análisis de datos. De esta manera la base de datos se hace portable en el caso de seguir monitoreando en otro computador o de realizar el análisis en otra estación de trabajo.

Gráfico D.51 Botones exportar e importar la base de datos (Proyecto).

La base de datos es creada y administrada por las aplicaciones de monitoreo y análisis de datos, esto permite que el usuario no instale paquetes adicionales para su administración.

La potencialidad de la base de datos se puede apreciar al recuperar los datos para el análisis estadístico.

D.1.6.2. Wireshark win32-1.2.8

Este software permite respaldar los datos en un archivo o en varios archivos de diferentes tamaños como Kilobytes, Megabytes, Gigabytes.

Incluye mecanismos para guardar los datos ya sea cada minuto, hora o día de la semana y para detener la captura de datos transcurrido cierto tiempo o una cantidad dada de paquetes.

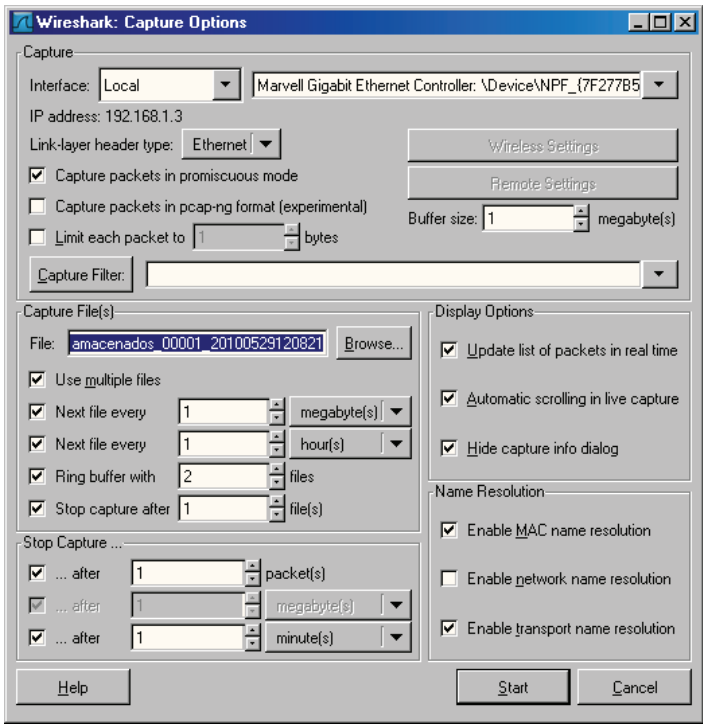


Gráfico D.52 Cuadro para respaldar archivos monitoreados (Wireshark).

El usuario también puede respaldar la captura de datos guardando todo el monitoreo en archivos con extensión .pcap los cuales se pueden abrir o compartir con otros tipos de aplicaciones, ya sean en Windows o Linux.

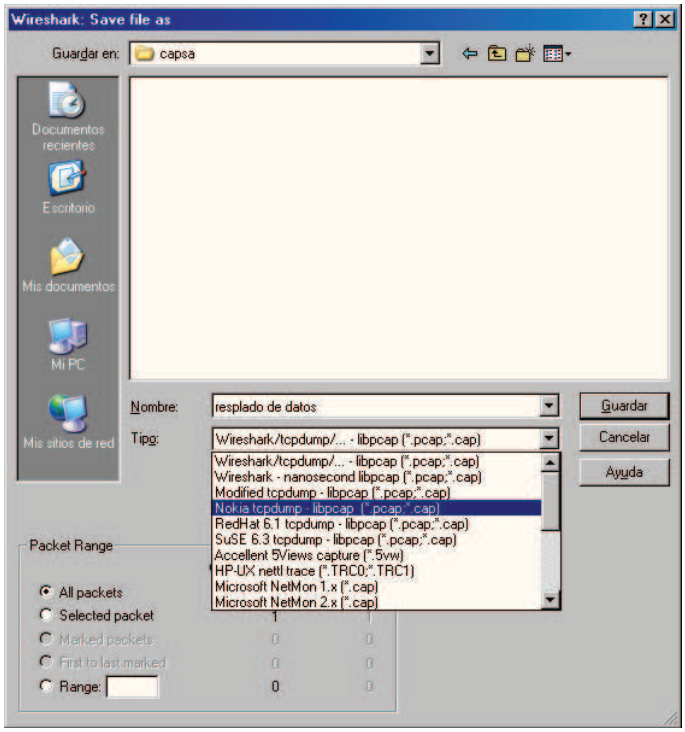


Gráfico D.53 Cuadro para respaldar la captura de datos (Wireshark).

D.1.6.3. Colasoft Capsa 7.1

La aplicación Capsa guarda la información monitoreada en uno o varios archivos contiguos con tamaños de Kilobytes, Megabytes y Gigabytes al igual que Wireshark ya que estos no disponen de una base de datos.

El software posee un buffer el cual almacena temporalmente los datos monitoreados para el cálculo de las gráficas estadísticas en tiempo real. Se puede asignar el tamaño del buffer para exportarlo o importarlo.

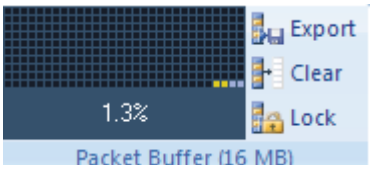


Gráfico D.54 Buffer de Paquetes (Capsa).

En la barra de herramientas existe un botón “Packet Storage” el cual permite abrir un cuadro de diálogo para asignar la forma de cómo guardar los datos monitoreados.

Gráfico D.55 Botón Packet Storage para asignar las opciones de guardado (Capsa).

El gráfico D.56 muestra un cuadro de diálogo que permite asignar el tamaño del buffer mencionado anteriormente, así como las opciones para almacenar todos los datos monitoreados.

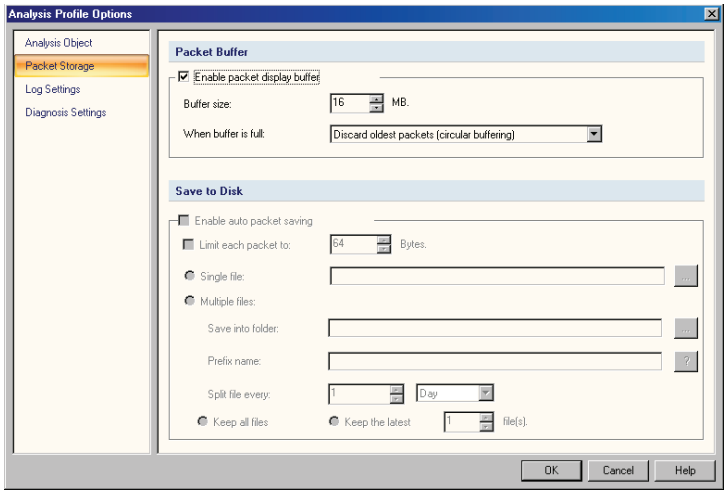


Gráfico D.56 Cuadro de diálogo para opciones de almacenamiento de paquetes de datos (Capsa).

Se puede guardar los datos con límites de tamaño para cada paquete, ya sea en un solo archivo o en múltiples archivos.

Los datos se pueden almacenar por intervalos de tiempo como meses, días, horas y minutos dependiendo del tiempo que el usuario requiera monitorear.

D.1.7. Obtención y Análisis de Datos almacenados

D.1.7.1. Proyecto de titulación

Para la obtención de datos se ha creado una aplicación independiente llamada “Query Statistics” que permite consultar los valores obtenidos de la base de datos.

Esta aplicación contiene una interfaz gráfica que ayuda al usuario a importar, exportar, conectar y desconectar la base de datos para realizar las consultas pertinentes al monitoreo realizado.

Gráfico D.57 Gráfico de la aplicación Query Statistics (Proyecto).

Al realizar la conexión a la base de datos se carga una lista con las opciones implementadas de análisis estadístico para los datos consultados.

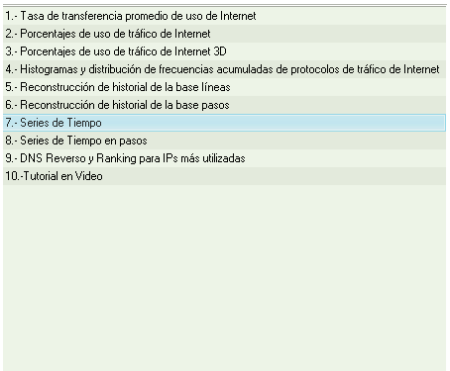


Gráfico D.58 Panel con lista de diagramas estadísticos (Proyecto).

Al seleccionar una opción en el cuadro anterior del gráfico D.58 se cargará una breve explicación del mismo en un área de texto.

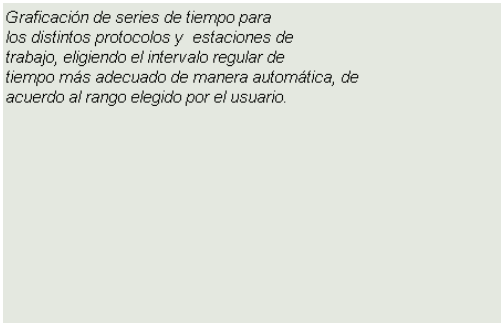


Gráfico D.59 Área de texto con descripción del diagrama seleccionado (Proyecto).

Además se adjunta una animación flash consiguiendo una visualización más elegante para el usuario sobre el tipo de análisis que eligió.

Gráfico D.60 Animación flash para diagrama seleccionado (Proyecto).

Para realizar cualquier tipo de análisis con los valores almacenados en la base de datos, se ha implementado un cuadro de diálogo; este permite elegir entre los protocolos TCP, datagrama UDP, mensajes ICMP o todos los protocolos, así como la fecha con horas y minutos de los hosts monitoreados y seleccionados por el usuario.

Por otro lado para realizar una consulta más específica sobre los datos monitoreados se añadió filtros para puertos y tipos de flujo de tráfico como son incoming y outgoing.

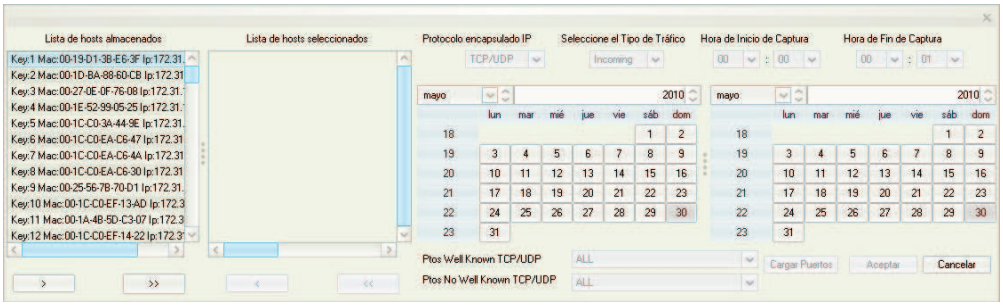


Gráfico D.61 Cuadro para selección de hosts, puertos e intervalo de tiempo para la consulta a la base de datos (Proyecto).

A continuación se detallará los tipos de gráficos que se obtendrán al consultar a la base de datos a través del cuadro de diálogo mostrado en el gráfico D.61, de esta manera se facilita la elección de los parámetros requeridos por el usuario.

D.1.7.1.1. Tasa de transferencia promedio de uso de Internet

Este tipo de diagrama permite visualizar el promedio de tasa de transferencia en las horas señaladas por el usuario, representando un rango de tiempo determinado.

Este promedio se lo realiza diferenciando estaciones de trabajo, protocolos y puertos.

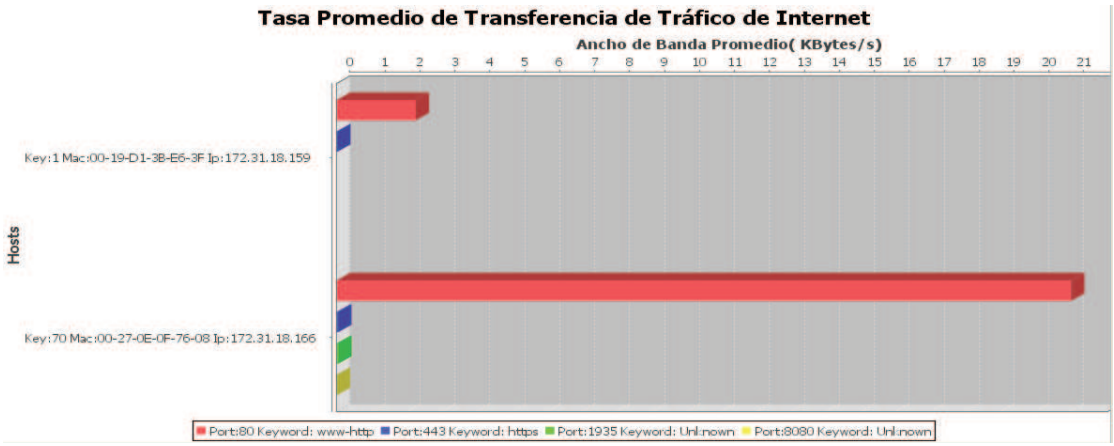


Gráfico D.62 Tasa de transferencia promedio de uso de Internet (Proyecto).

Para más detalle se añade un área de texto con los valores promedio de la tasa de transferencia de tráfico de Internet para los hosts seleccionados.

```
---------HOST: Key:1 Mac:00-19-D1-3B-E6-3F Ip:172.31.18.159---------
Port:80 Keyword: www-http Promedio: 2.260420480387057 KBytes/s
Port:443 Keyword: https Promedio: 0.012288182218309 86 KBytes/s
---------HOST: Key:70 Mac:00-27-0E-0F-76-08 Ip:172.31.18.166---------
Port:80 Keyword: www-http Promedio: 21.017752514137857 KBytes/s
Port:1935 Keyword: Unknown Promedio: 0.042626869764991464 KBytes/s
Port:443 Keyword: https Promedio: 0.037455506595443874 KBytes/s
Port:8080 Keyword: Unknown Promedio: 1.0465849071702944E-4 KBytes/s
```

Gráfico D.63 Área de texto con los valores de Tasa de transferencia promedio de uso de Internet (Proyecto).

Este tipo de diagrama posee un botón de ayuda que permite cargar una página web, la cual mostrará la forma de interpretar los datos obtenidos.

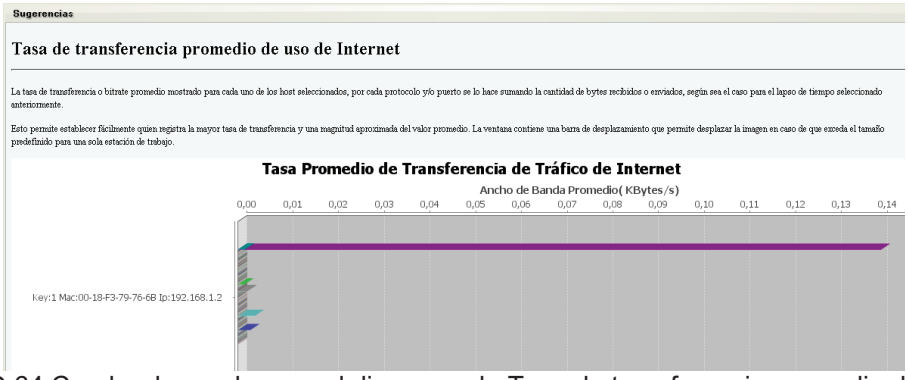


Gráfico D.64 Cuadro de ayuda para el diagrama de Tasa de transferencia promedio de uso de Internet (Proyecto).

D.1.7.1.2. Porcentajes de uso de tráfico de Internet

Este tipo de diagrama permite obtener valores porcentuales sobre el consumo de tráfico de Internet, especificando estaciones de trabajo, protocolos de capa transporte, y protocolos de capa aplicación monitoreados en un rango de tiempo.

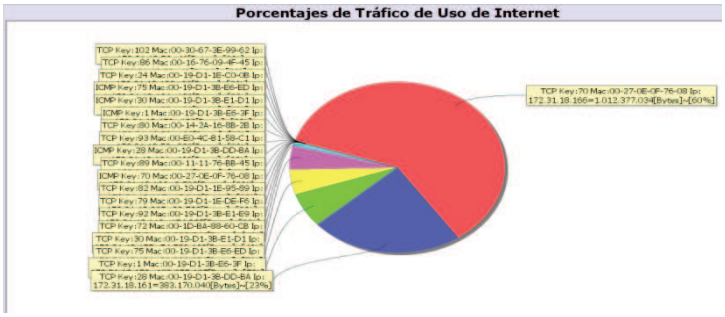


Gráfico D.65 Porcentajes de uso de tráfico de Internet (Proyecto).

Los detalles de los valores del diagrama representado por el pastel del gráfico D.65 se localizan en el siguiente cuadro de texto.

TCP Key:70 Mac:00-27-0E-0F-76-08 Ip:172.31.18.166
Tamaño Bytes: 1.012377034E9 Porcentaje: 60.456026293531394
TCP Key:28 Mac:00-19-D1-3B-DD-BA Ip:172.31.18.161
Tamaño Bytes: 3.8317004E8 Porcentaje: 22.881730062175112
TCP Key:1 Mac:00-19-D1-3B-E6-3F Ip:172.31.18.159
Tamaño Bytes: 1.09055107E8 Porcentaje: 6.512433801650107
TCP Key:75 Mac:00-19-D1-3B-E6-ED Ip:172.31.18.164
Tamaño Bytes: 8.0286954E7 Porcentaje: 4.794488652980986
TCP Key:30 Mac:00-19-D1-3B-E1-D1 Ip:172.31.18.155
Tamaño Bytes: 7.4539169E7 Porcentaje: 4.451248704405103
TCP Key:72 Mac:00-1D-BA-88-60-CB Ip:172.31.18.34
Tamaño Bytes: 1.4933344E7 Porcentaje: 0.8917731311498215
TCP Key:92 Mac:00-19-D1-3B-E1-E9 Ip:172.31.18.112
Tamaño Bytes: 154835.0 Porcentaje: 0.009246267464379218
TCP Key:79 Mac:00-19-D1-1E-DE-F6 Ip:172.31.18.205
Tamaño Bytes: 29532.0 Porcentaje: 0.0017635597297642463
TCP Key:82 Mac:00-19-D1-1E-95-89 Ip:172.31.18.117
Tamaño Bytes: 17040.0 Porcentaje: 0.0010175761138826614
ICMP Key:70 Mac:00-27-0E-0F-76-08 Ip:172.31.18.166

Gráfico D.66 Área de texto con los valores de Tasa Porcentajes de uso de tráfico de Internet (Proyecto).

El gráfico posee dos botones que permiten girar el pastel en sentido horario o anti horario.

Gráfico D.67 Botones para girar el pastel (Proyecto).

El cuadro de diálogo incluye un botón para acceder al contenido de ayuda para la interpretación de los datos obtenidos.

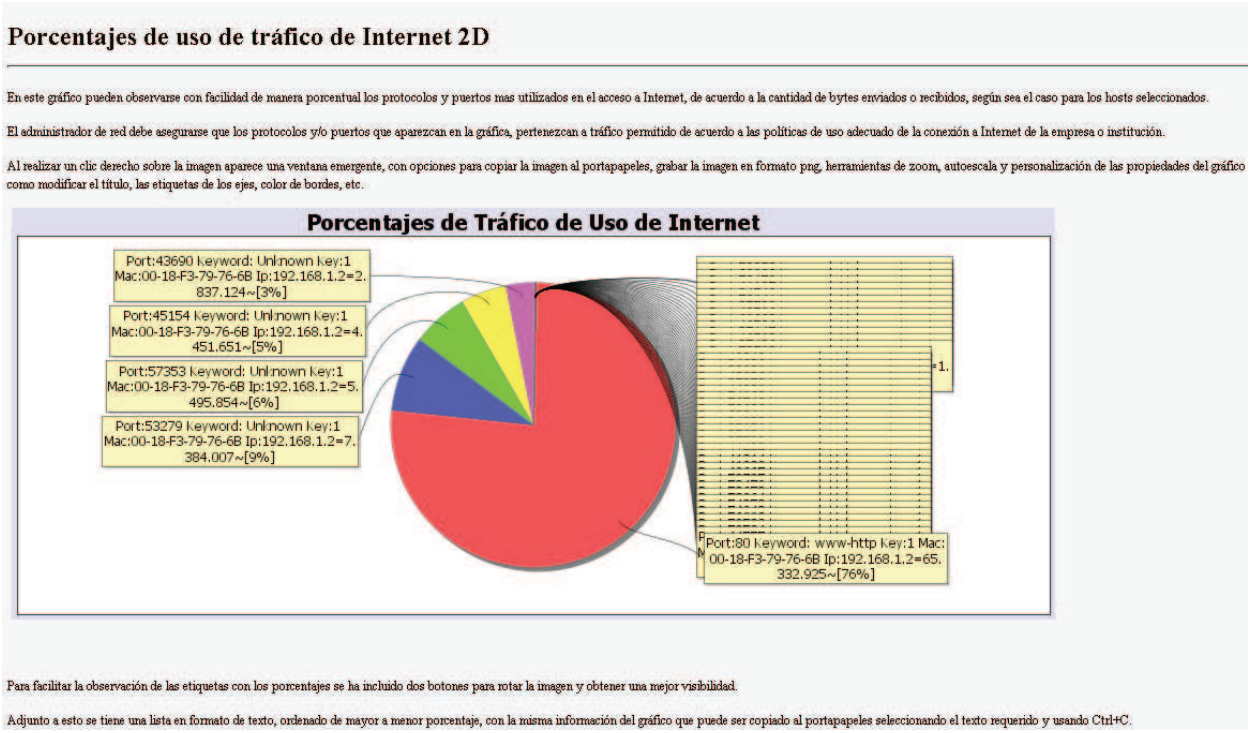


Gráfico D.68 Cuadro de ayuda para el diagrama de Porcentajes de uso de tráfico de Internet (Proyecto).

D.1.7.1.3. Porcentajes de uso de tráfico de Internet 3D.

Esta representación gráfica permite obtener valores porcentuales en tres dimensiones sobre el consumo de tráfico de Internet. Además permite mostrar con más detalle la minoría de los datos en el pastel, resultado de filtrar el tráfico monitoreado por estaciones de trabajo, protocolos de capa transporte, protocolos de capa aplicación para un intervalo de tiempo.

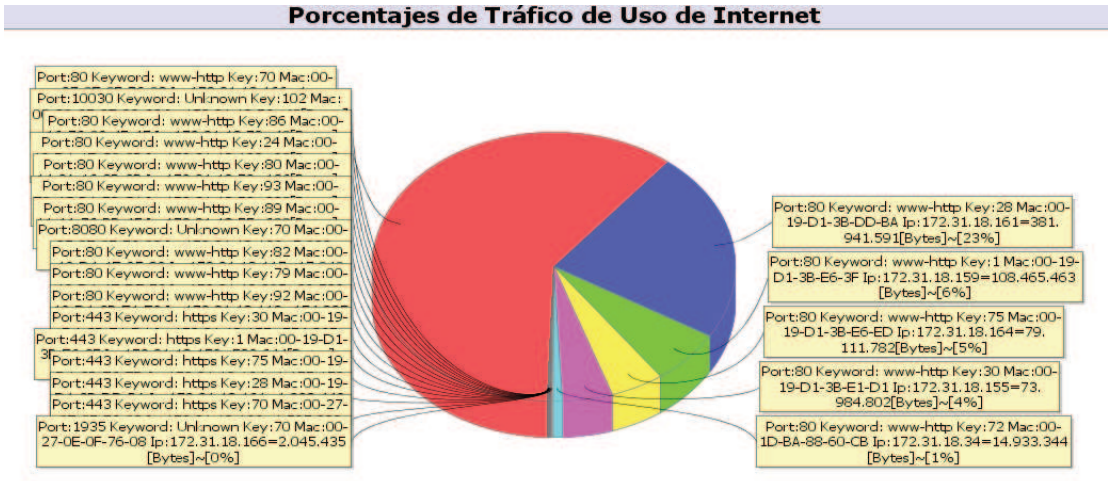


Gráfico D.69 Porcentajes de uso de tráfico de Internet 3D (Proyecto).

Para los valores casi imperceptibles mostrados en el pastel del tráfico monitoreado, se añade un área de texto para representar con más detalle dichos datos.

Gráfico D.70 Área de texto con los valores de Tasa Porcentajes de uso de tráfico de Internet 3D (Proyecto).

El gráfico D.71 muestra dos botones que permiten girar el pastel 3D en sentido horario y anti horario.

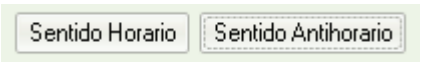


Gráfico D.71 Botones para girar el pastel 3D (Proyecto).

Como se muestra en el gráfico D.72 incluye una ayuda que permite interpretar de mejor manera los resultados.

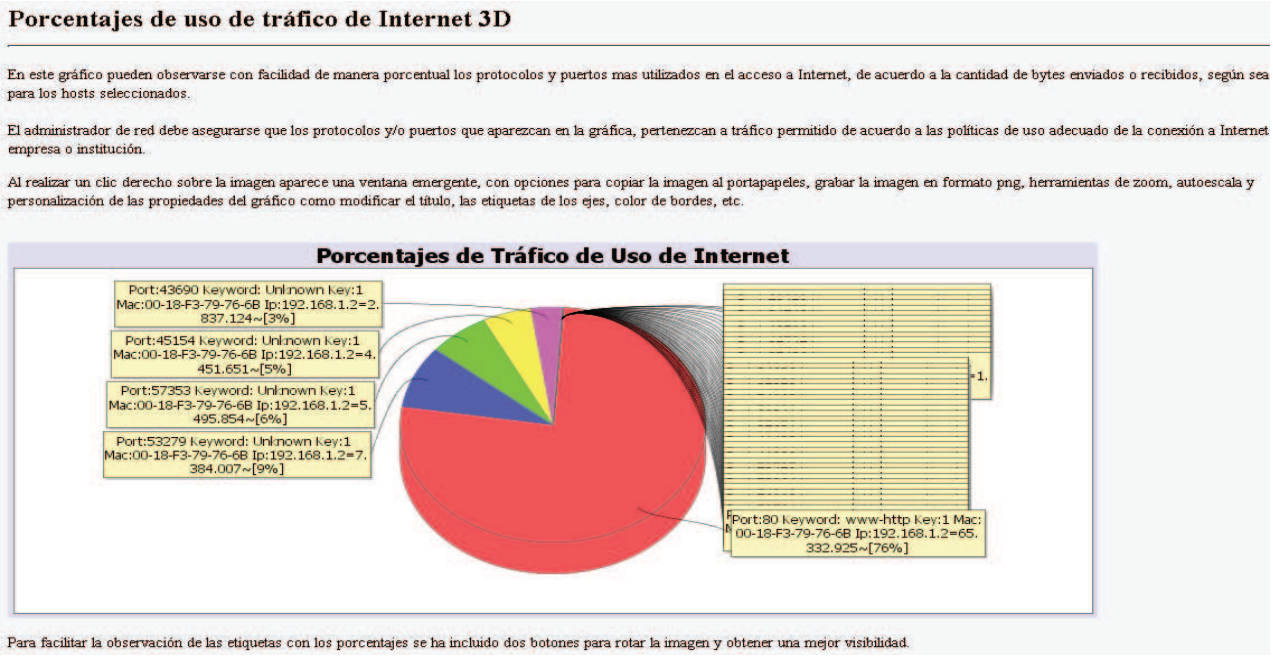


Gráfico D.72 Cuadro de ayuda para el diagrama de Porcentajes de uso de tráfico de Internet 3D (Proyecto).

D.1.7.1.4. Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet.

Esta opción genera un histograma de frecuencias con los datos obtenidos de la base, estableciendo de manera automática rangos uniformes para los valores de ancho de banda de tráfico de Internet monitoreados.

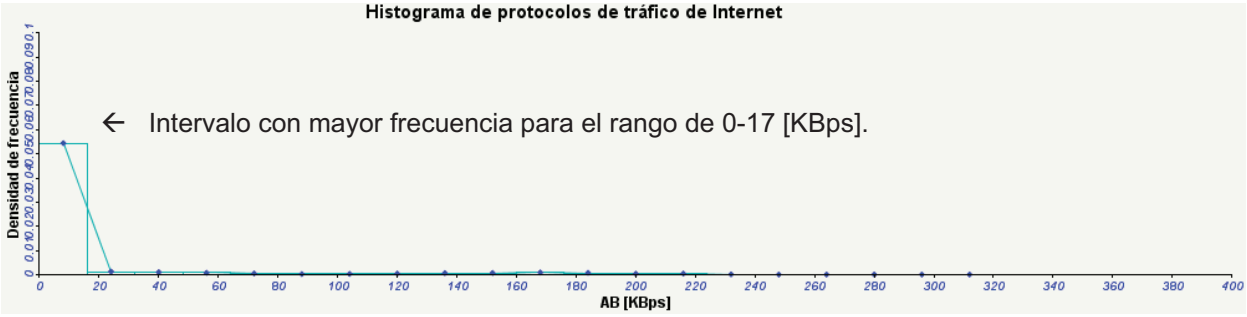


Gráfico D.73 Histograma de protocolos de tráfico de Internet (Proyecto).

Conjuntamente se grafica la distribución de frecuencias acumuladas para los valores obtenidos durante el monitoreo del tráfico de Internet correspondiente al gráfico D.74.

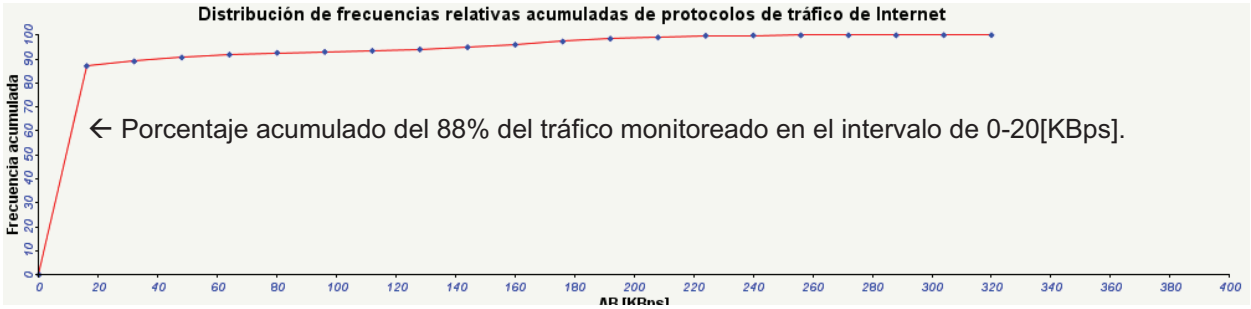


Gráfico D.74 Distribución de frecuencias relativas acumuladas (Proyecto).

El número de intervalos para los gráficos anteriores se puede cambiar dependiendo del detalle que requiera el usuario para la interpretación de los datos obtenidos.

Gráfico D.75 Número de intervalos para histograma y distribución de frecuencias (Proyecto).

Los valores resultantes de la consulta a la base de datos para los diagramas anteriores son representados por una tabla de frecuencias, la cual permite clasificar los valores de tasa de transferencia respecto a su magnitud dentro de los intervalos, frecuencia relativa y densidad.

Tabla de Frecuencias

Intervalo de Clase (KBps)	Frecuencia	Frecuencia relativa	Densidad
0.0-<16.000000000000004	1150	0,870553	0,0544095
16.000000000000004-<32.00000000000001	26	0,0196821	0,00123013
32.00000000000001-<48.000000000000014	21	0,015897	0,000993565
48.000000000000014-<64.00000000000001	16	0,012112	0,000757002
64.00000000000001-<80.00000000000001	10	0,00757002	0,000473126

Gráfico D.76 Tabla de Frecuencias (Proyecto).

También se incluye una tabla de resumen de estadística descriptiva, la cual contiene información sobre el número de valores de bitrate, la media, el error estándar de la media, la media recortada, la desviación estándar, además muestra el valor mínimo, el máximo e indicadores como el primer cuartil, la mediana (segundo cuartil) y el tercer cuartil.

Tabla de Estadística Descriptiva

Variable	N	Media	Error estándar de la media	Media recortada (5%)	Desviación estándar
Bitrate	1.321	14,819	1,22264	6,38602	44,4374

Variable	Valor mínimo	Q1	Mediana	Q3	Valor máximo
Bitrate	0	0	0	0	300,492

Gráfico D.77 Tabla de Estadística Descriptiva (Proyecto).

Se ha añadido una ayuda de interpretación de resultados para el usuario.

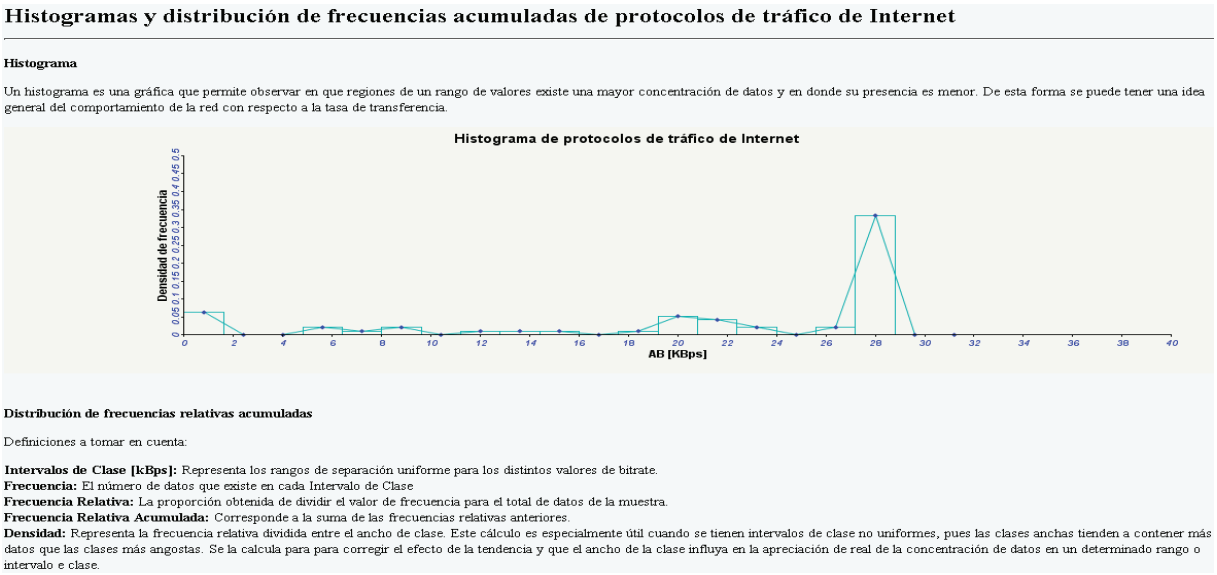


Gráfico D.78 Cuadro de ayuda para el diagrama de Histograma y distribución de frecuencias acumuladas de protocolos de tráfico de Internet (Proyecto).

D.1.7.1.5. Reconstrucción de historial de la base de datos con líneas.

Por medio de un diagrama de líneas se reconstruirán los valores de la base de datos del tráfico de Internet entrante o saliente, para las estaciones de trabajo, protocolos y rango de tiempo seleccionados.

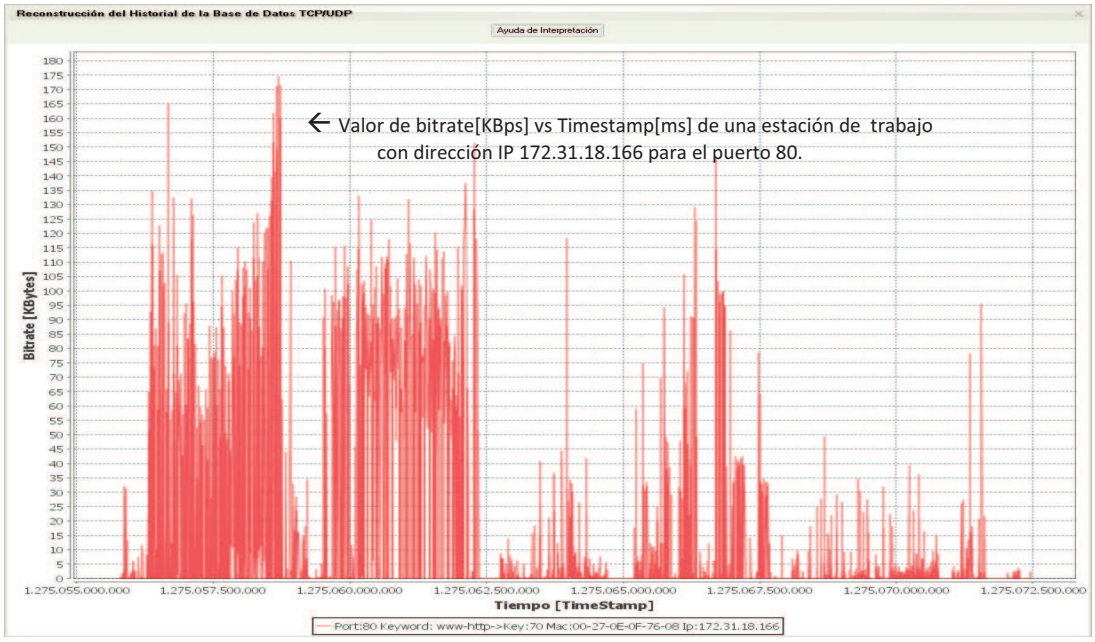


Gráfico D.79 Reconstrucción de historial de la base de datos con líneas (Proyecto).

Como se muestra en el gráfico D.80 se incluye una ayuda de interpretación de datos para el usuario, de esta manera se podrá interpretar con más facilidad los valores obtenidos al realizar el monitoreo de la red.

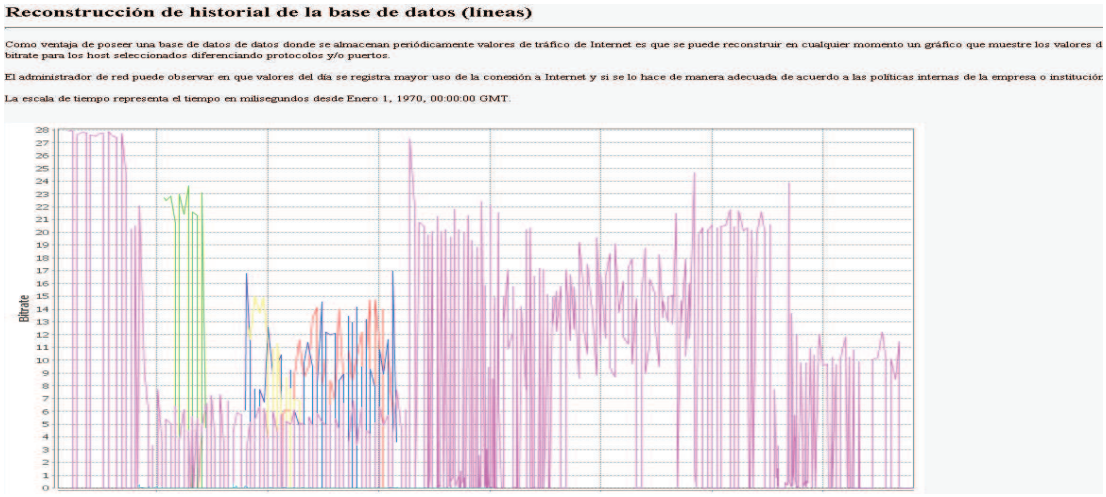


Gráfico D.80 Cuadro de ayuda para el diagrama de Reconstrucción de historial de la base de datos con líneas (Proyecto).

D.1.7.1.6. Reconstrucción de historial de la base de datos en pasos.

Realiza una representación gráfica en el tiempo de los valores de bitrate recuperados de la base de datos, representados en pasos. Estos valores son filtrados por protocolos, puertos y estaciones de trabajo de acuerdo al rango de tiempo elegido por el usuario.

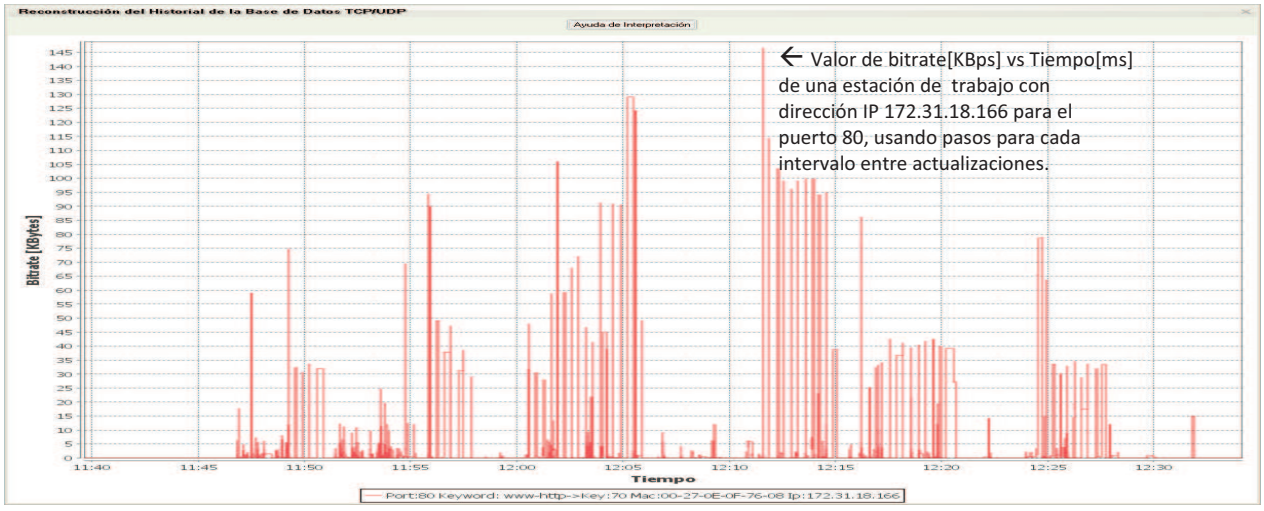


Gráfico D.81 Reconstrucción de historial de la base de datos en pasos (Proyecto).

Además el diagrama posee una ayuda que permite interpretar de mejor manera este diagrama como se muestra en el gráfico D.82.

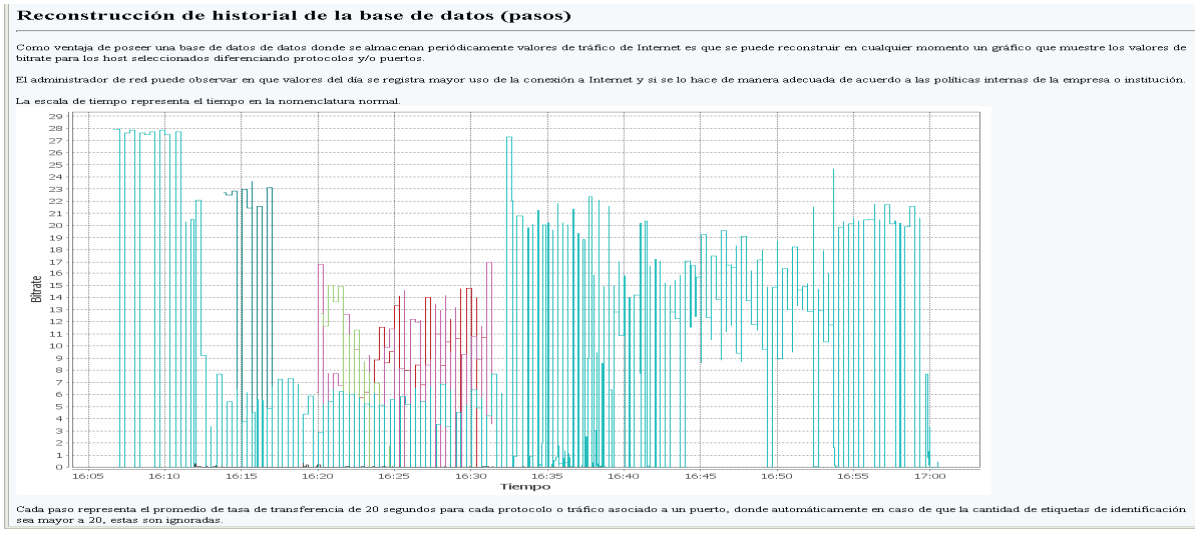


Gráfico D.82 Cuadro de ayuda para el diagrama de Reconstrucción historial de la base de datos en pasos (Proyecto).

D.1.7.1.7. Series de tiempo

Grafica series de tiempo para los distintos protocolos y estaciones de trabajo, asignando el intervalo de tiempo más adecuado para la representación de los datos monitoreados de manera automática, de acuerdo al rango de tiempo elegido por el usuario. Además permite obtener información de la actividad de la estación de trabajo visualizándolo de manera gráfica.

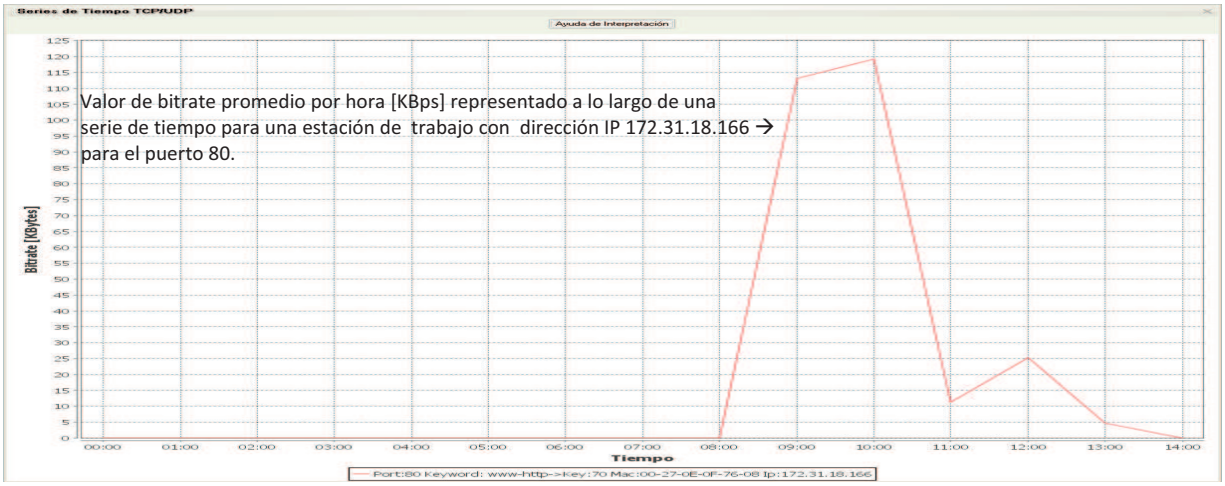


Gráfico D.83 Series de tiempo (Proyecto).

La ayuda añadida para este diagrama será una pieza fundamental para los criterios de análisis del usuario al interpretar los datos como se muestra en el gráfico D.84.

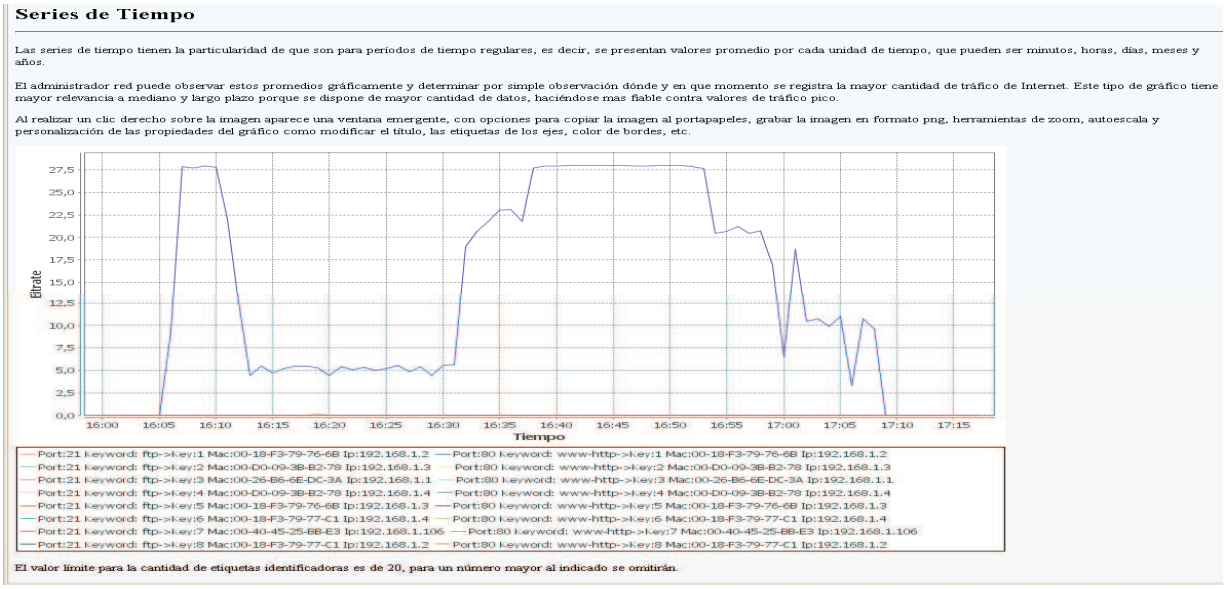


Gráfico D.84 Cuadro de ayuda para el diagrama Series de tiempo (Proyecto).

D.1.7.1.8. Series de tiempo en pasos.

Grafica series de tiempo en pasos asignando el intervalo de tiempo más adecuado para la representación de los datos monitoreados de manera automática, de acuerdo al rango de tiempo elegido por el usuario para cada protocolo y estación de trabajo. Además permite obtener información de la actividad de la estación de trabajo en un determinado espacio de tiempo.

Esto permite mostrar en qué momento una o un grupo de estaciones de trabajo ha tenido mayor actividad en un intervalo de tiempo.

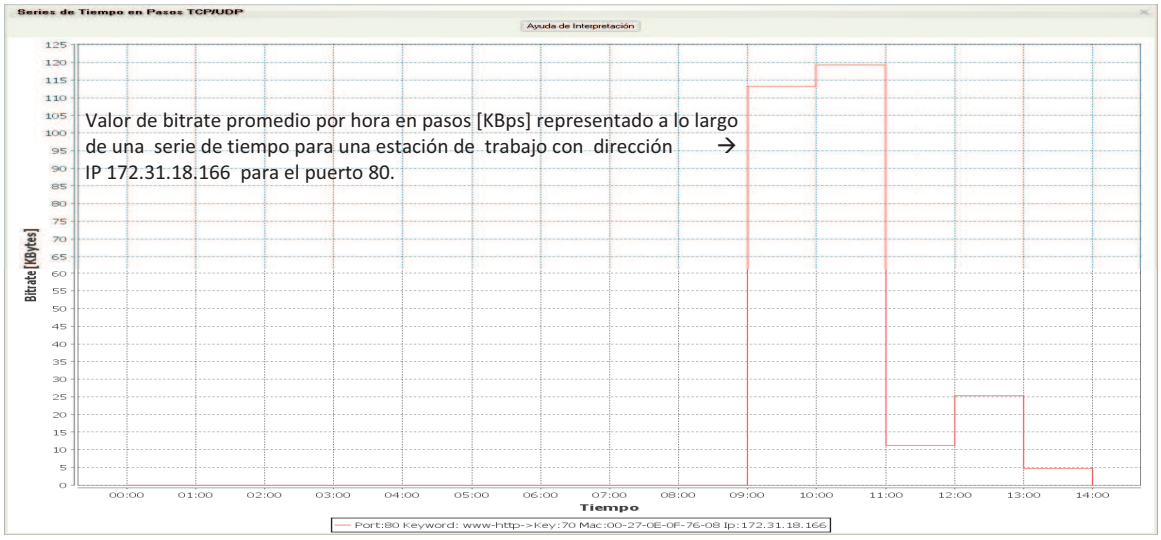


Gráfico D.85 Series de tiempo en pasos (Proyecto).

La ayuda de interpretación correspondiente al gráfico D.85 se muestra través del botón situado en la parte superior del mismo.

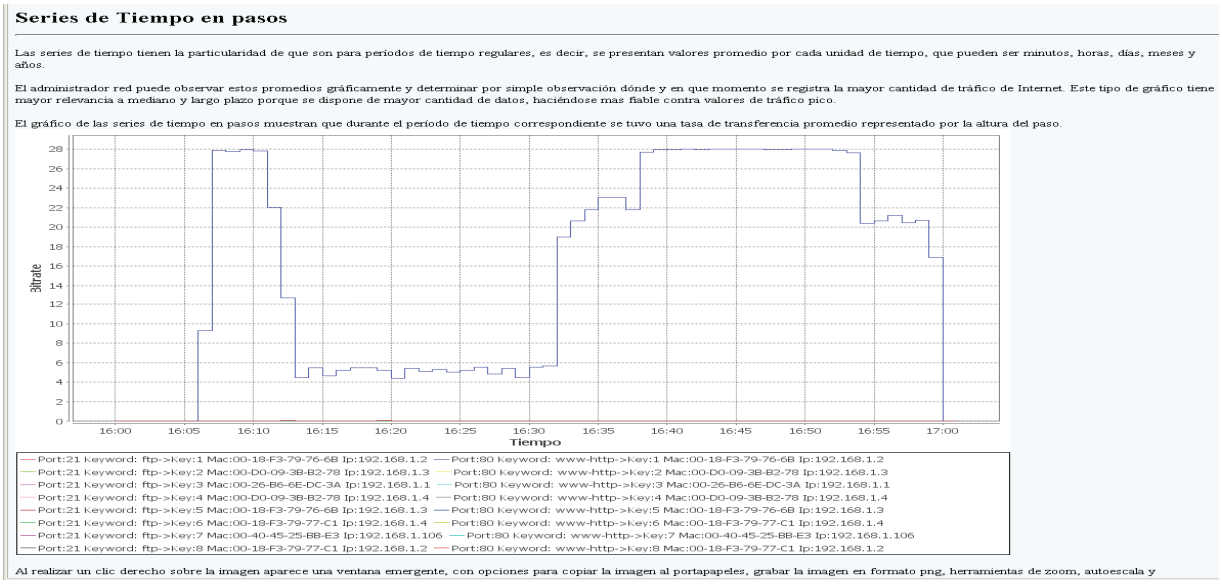
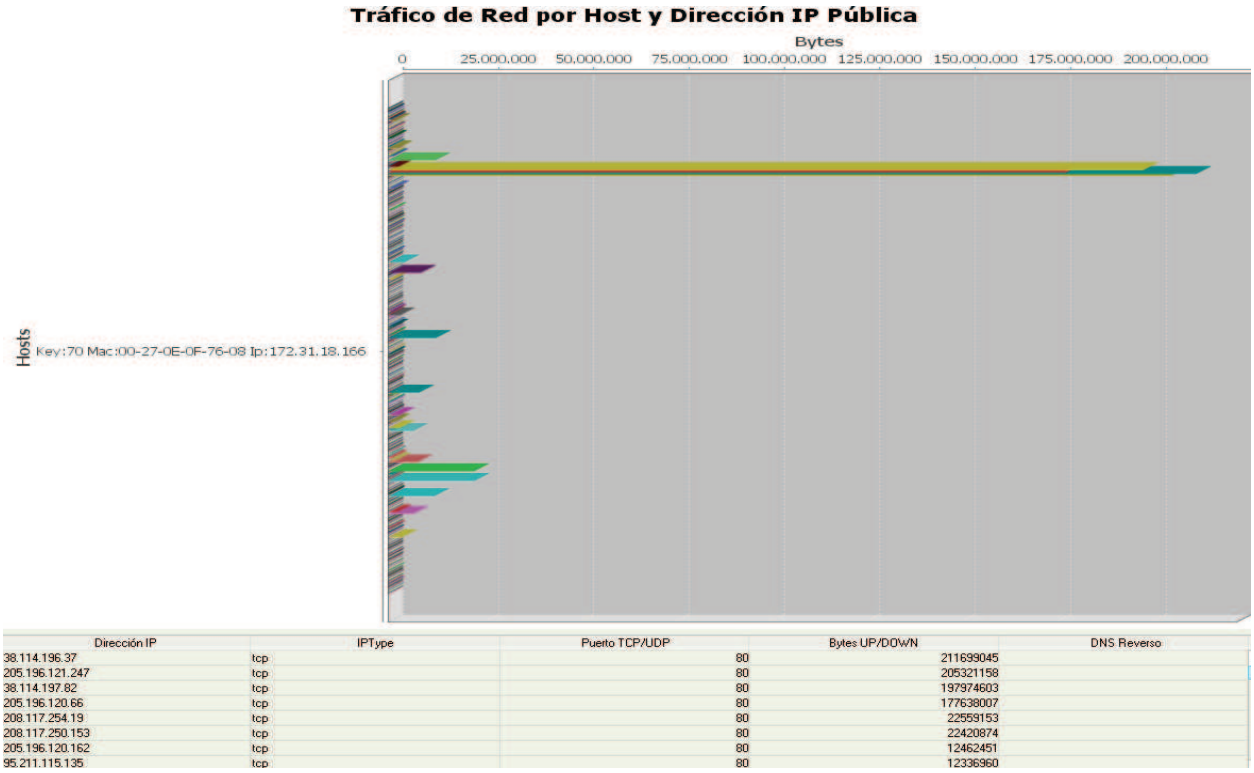


Gráfico D.86 Cuadro de ayuda para el diagrama Series de tiempo en pasos (Proyecto).

D.1.7.1.9. DNS reverso y ranking para IPs más utilizadas

Utilizando un diagrama de barras se mostrará la cantidad de bytes transferidos para las distintas direcciones IP registradas en la base de datos por cada host seleccionado. Incluye un mecanismo que realiza el proceso de DNS reverso para la resolución de nombres de dominio de las direcciones IP.



Dirección IP	IPType	Puerto TCP/UDP	Bytes UP/DOWN	DNS Reverso
38.114.196.37	tcp	80	211699045	
205.196.121.247	tcp	80	205321158	
38.114.197.82	tcp	80	197974603	
205.196.120.66	tcp	80	177638007	
208.117.254.19	tcp	80	22559153	
208.117.250.153	tcp	80	22420874	
205.196.120.162	tcp	80	12462451	
95.211.115.135	tcp	80	12336960	

Gráfico D.87 DNS reverso y ranking para IPs más utilizadas (Proyecto).

El software permite asignar uno o varios DNS predeterminados o en caso contrario el usuario podrá ingresar el DNS que desee para la resolución de nombres de dominio. Por otro lado se añadió un botón mostrado en el gráfico D.88 para iniciar un navegador web y comprobar las direcciones IP monitoreadas.

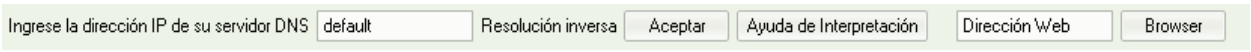


Gráfico D.88 Cuadro de texto para ingresar servidor DNS (Proyecto).

Se ha incluido una ayuda para una mejor interpretación de parte del usuario como se muestra en el gráfico D.89.

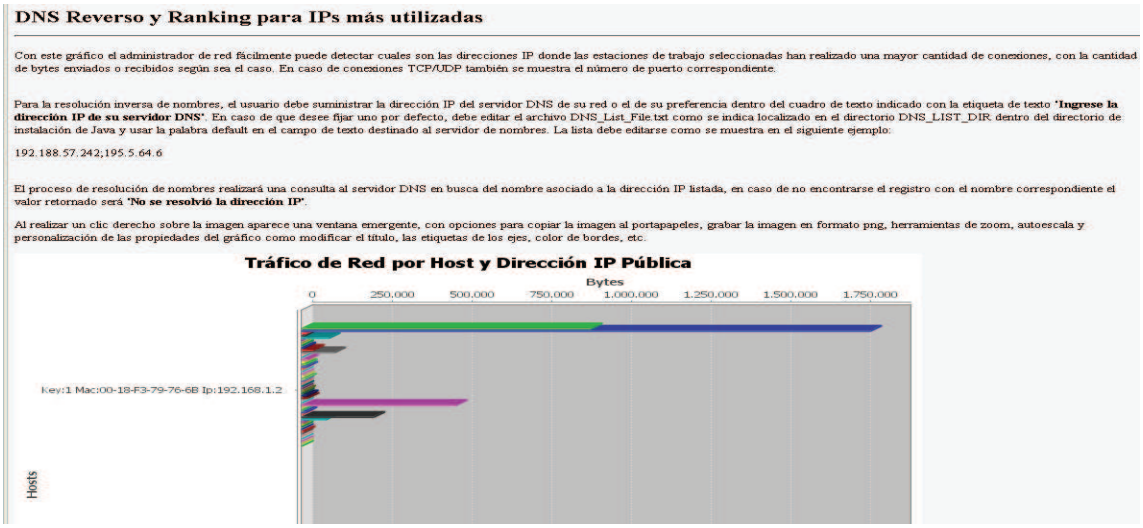


Gráfico D.89 Cuadro de ayuda para el diagrama D.87 (Proyecto).

D.1.7.1.10. Tutorial en Video

Reproduce un video explicativo del funcionamiento del programa.

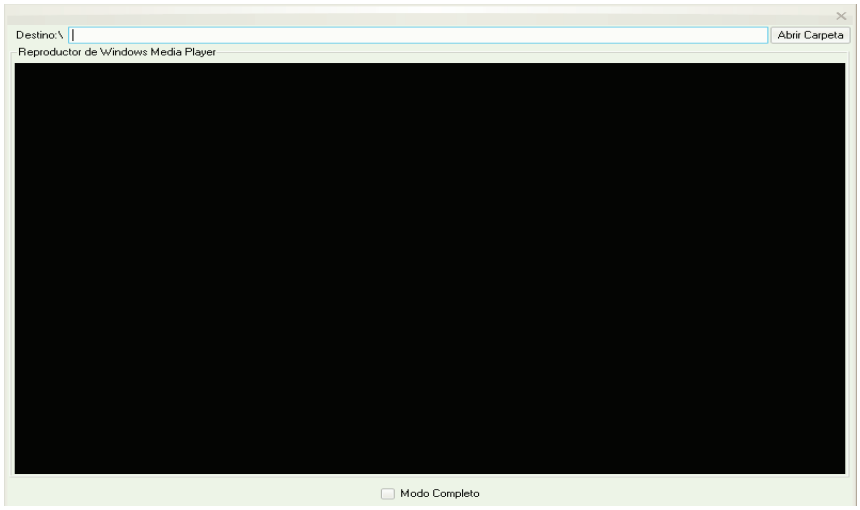


Gráfico D.90 Tutorial en Video (Proyecto).

D.1.7.2. Wireshark win32-1.2.8

Para el análisis de datos Wireshark puede cargar los respaldos del tráfico monitoreado desde cualquier carpeta a través del cuadro diálogo mostrado en el gráfico D.91.

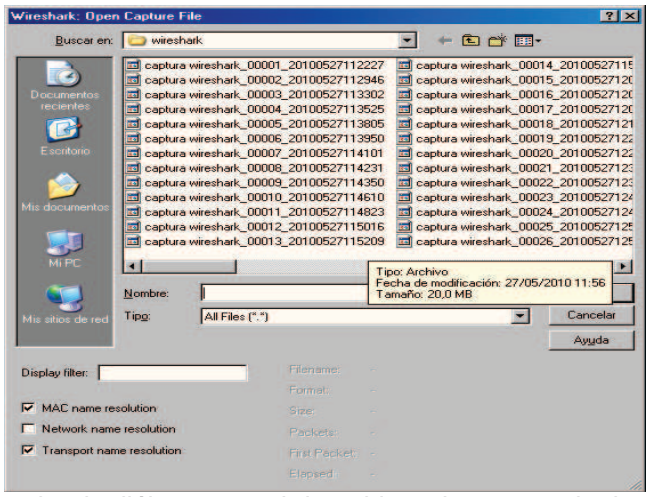


Gráfico D.91 Cuadro de diálogo para abrir archivos de captura de datos (Wireshark).

Después de haber seleccionado los archivos necesarios para el análisis de datos el software carga todos los valores del tráfico capturado en la pantalla principal.

Este software incluye herramientas para poder analizar la estructura y valores de los paquetes monitoreados.

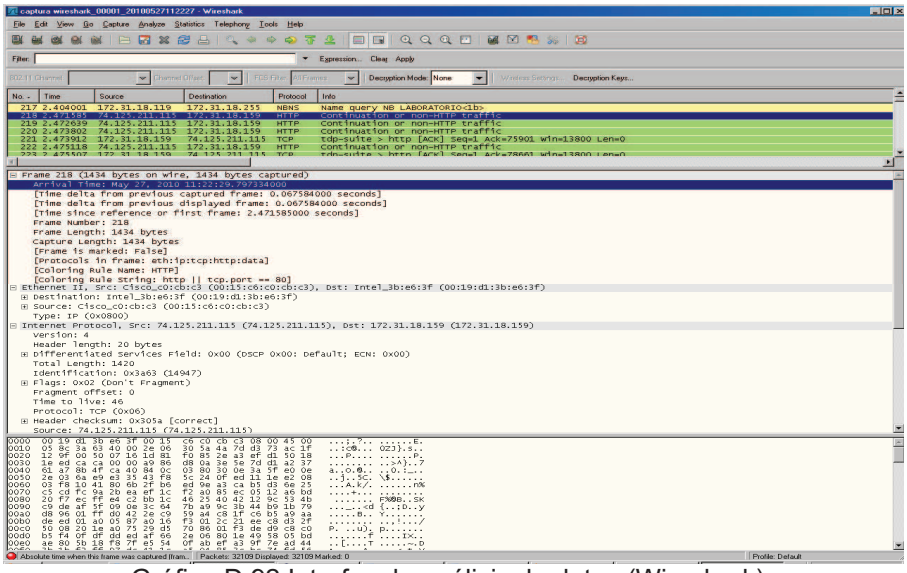


Gráfico D.92 Interfaz de análisis de datos (Wireshark).

Además permite analizar la cantidad de paquetes enviados y recibidos entre los hosts filtrados y las direcciones IP con las que se ha conectado.

El gráfico D.93 muestra un cuadro de diálogo que permite analizar el tráfico monitoreado a nivel de direcciones IPv4, paquetes TCP, datagramas UDP y tráfico Ethernet.

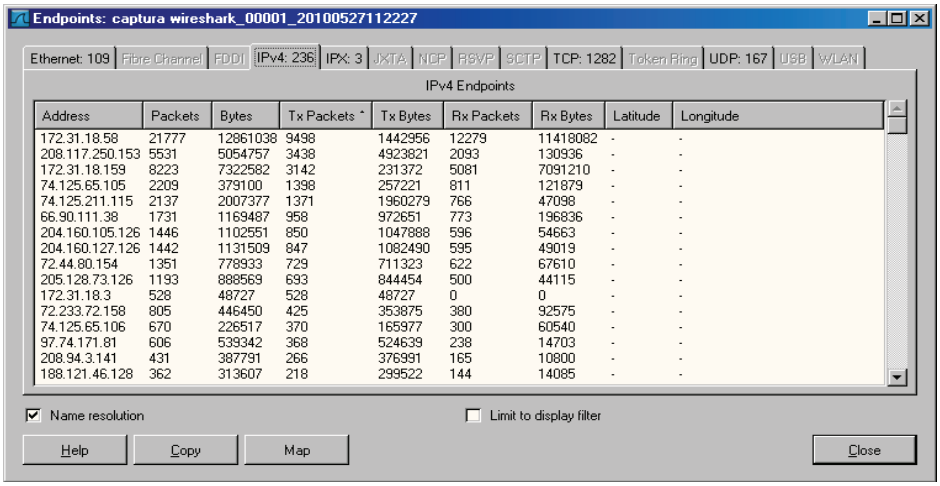


Gráfico D.93 Cuadro de diálogo de análisis por protocolo para los datos monitoreados (Wireshark).

También es posible visualizar gráficamente las conversaciones con las peticiones y respuestas entre el host monitoreado y las direcciones IP con las que se ha conectado.

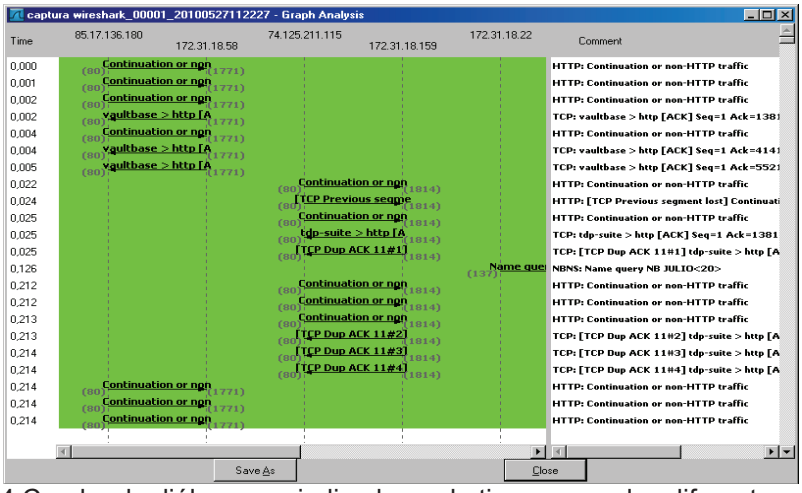


Gráfico D.94 Cuadro de diálogo con indicadores de tiempo para las diferentes peticiones y respuestas (Wireshark).

Wireshark al ser un analizador de protocolos de red solo permite realizar filtros y mostrar los diferentes valores de los datos transferidos. Esta es la razón por la que es muy útil para solucionar problemas en redes de comunicaciones para desarrollo de software y protocolos, además aporta como una herramienta didáctica.

D.1.7.3. Colasoft Capsa 7.1

Para recuperar los archivos guardados del monitoreo de datos se selecciona en la pantalla de inicio la pestaña “Replay”, la cual le permitirá añadir el archivo que se va a analizar.

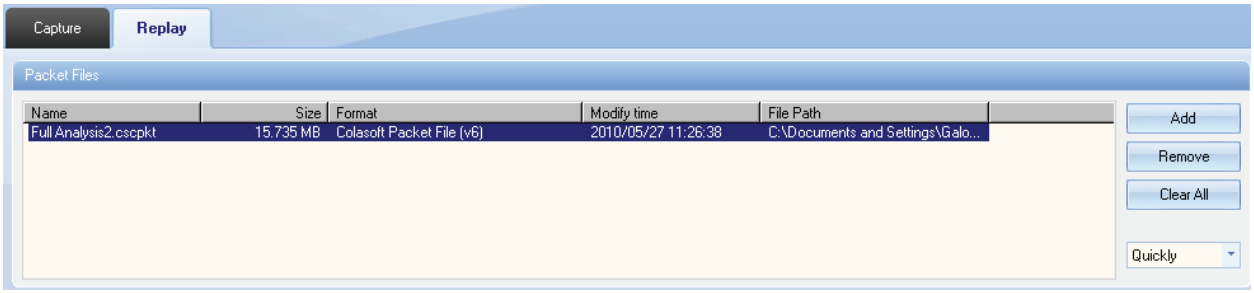


Gráfico D.95 Pestaña para recuperar los archivos guardados del monitoreo de datos (Capsa).

Para el análisis de la captura, el software incluye múltiples pestañas la cuales permiten la selección de un host específico, mostrando los resultados acerca del tráfico que ha generado dicha estación de trabajo.

La primera pestaña “Dashboard” mostrada en el gráfico D.96 es habilitada cuando se selecciona el tráfico total monitoreado y genera gráficos de la tasa de transferencia.

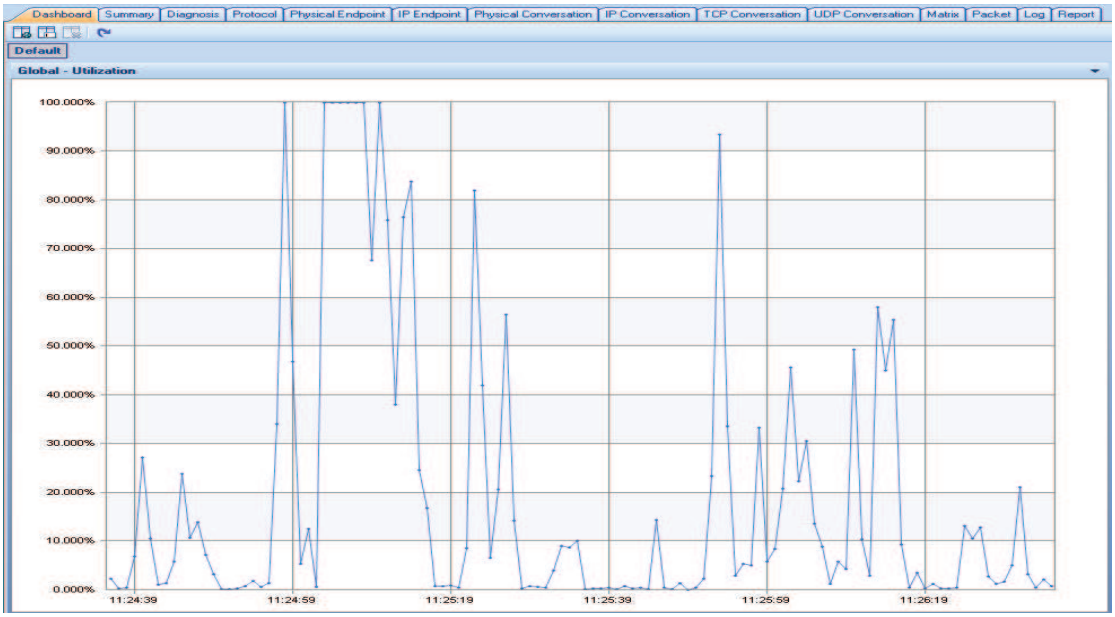


Gráfico D.96 Porcentaje utilización del tráfico de Internet (Capsa).

Para la utilización del resto de pestañas se necesita elegir una estación de trabajo como se describe en el gráfico D.97.

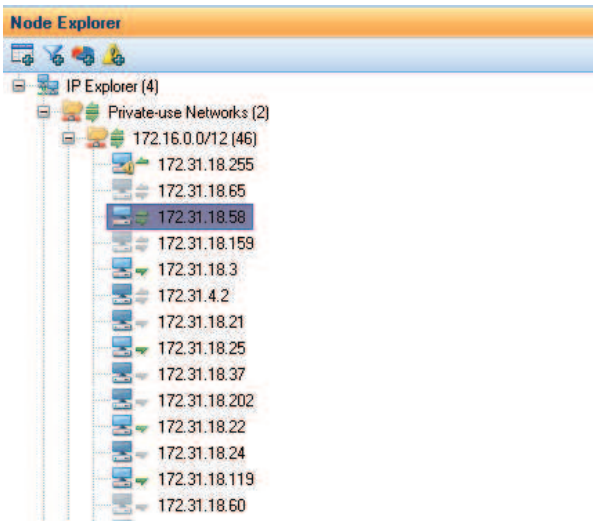


Gráfico D.97 Lista de hosts monitoreados (Capsa).

Seleccionando un host se habilitan las pestañas para el análisis de datos, siendo la primera pestaña “Summary” descrita en el gráfico D.98. Esta muestra el tráfico total de los datos transferidos por este host, así como las conversaciones TCP, UDP, IP y análisis DNS, Email, Ftp, Http.

Summary | Diagnosis | Protocol | IP Endpoint | IP Conversation | TCP Conversation | UDP Conversation | Matrix | Packet | Log | Report

Statistics Item					Current Value
Traffic	Bytes	Packets	Utilization	Bits Per Second	Packets Per Se...
Total	9.602 MB	16,421	0.000%	0 bps	0
Broadcast	0 B	0	0.000%	0 bps	0
Multicast	0 B	0	0.000%	0 bps	0
Receive	8.405 MB	8,949	-	-	-
Sent	1.197 MB	7,472	-	-	-
Flow					Count
IP Conversation					98
TCP Conversation					746
UDP Conversation					5
TCP					Count
TCP SYN Sent					770
TCP SYN Received					0
TCP SYNACK Sent					0
TCP SYNACK Received					1,015
TCP FIN Sent					229
TCP FIN Received					362
TCP Reset Sent					459
TCP Reset Received					16
DNS Analysis					Count
DNS Query					166
DNS Response					0
Email Analysis					Count
SMTP Connection					0
POP3 Connection					0
FTP Analysis					Count
FTP Upload					0
FTP Download					0
HTTP Analysis					Count
HTTP Request					0
HTTP Connection					0

Gráfico D.98 Pestaña Summary (Capsa).

La pestaña “Diagnosis” del gráfico D.99 permite mostrar todos los diagnósticos de conexión de capa aplicación, transporte, red así como una lista de eventos generados por el host seleccionado.

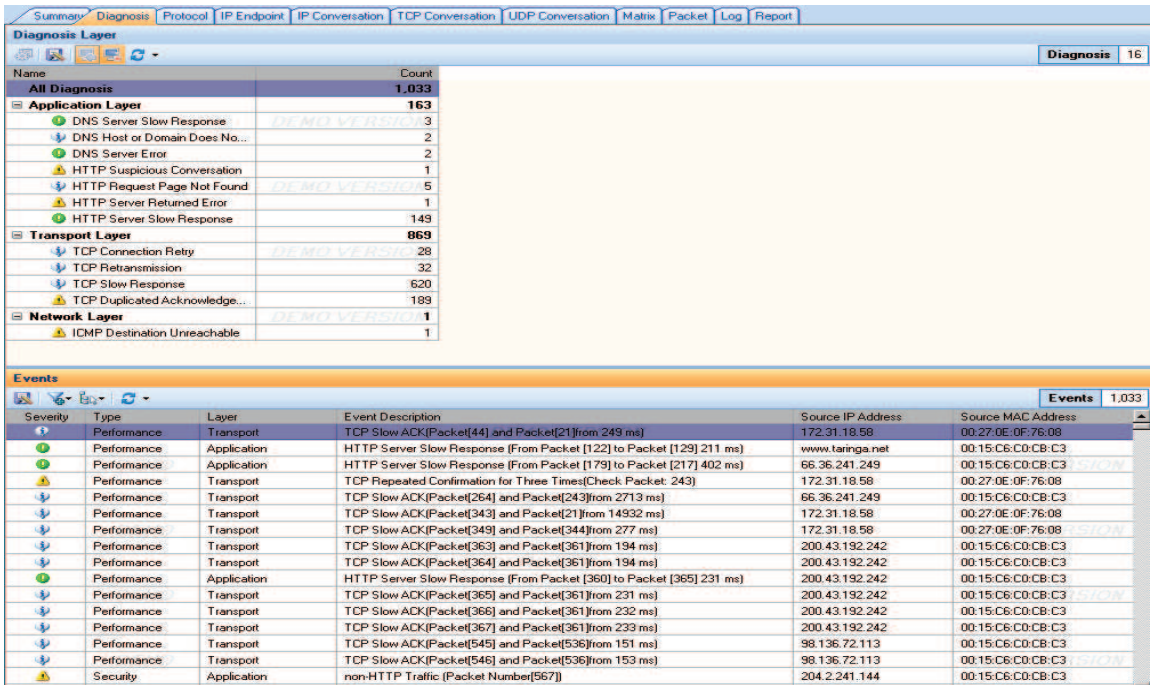


Gráfico D.99 Pestaña Diagnosis (Capsa).

La pestaña “Protocol” del gráfico D.100 permite mostrar los protocolos que el host seleccionado ha utilizado para la transferencia de información, para ello se muestra una tabla con valores de bytes y paquetes enviados y recibidos.

La tabla que se encuentra en la parte de inferior del gráfico D.100 corresponde a conversaciones IP, TCP y UDP que son las mismas opciones que se muestra en las pestañas de la parte superior del mismo gráfico y se detallarán más adelante.

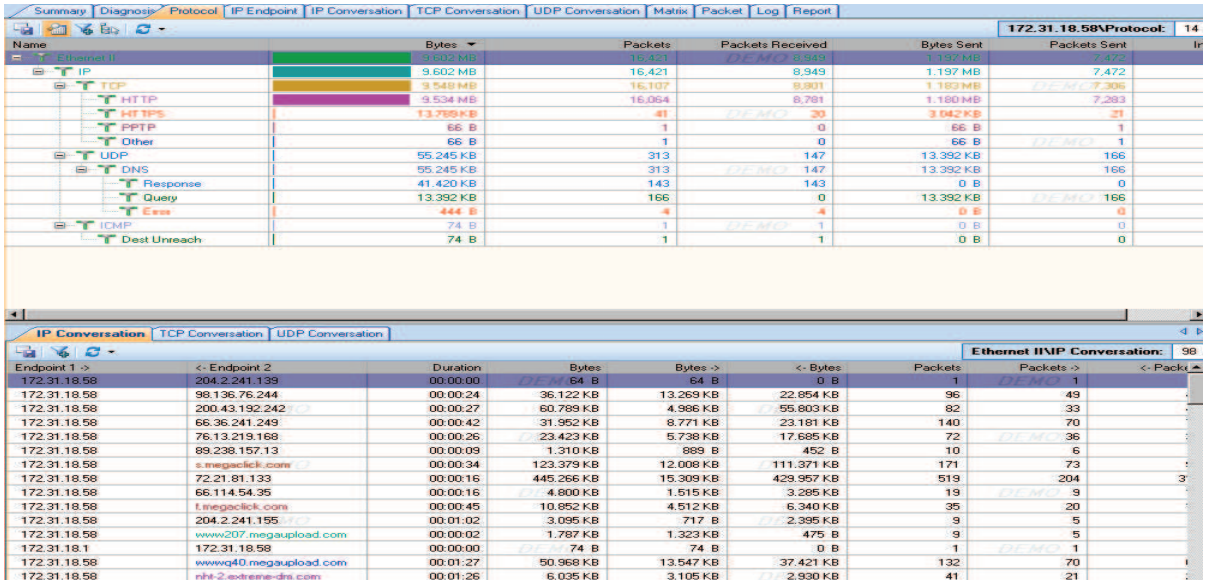


Gráfico D.100 Pestañas Protocol e IP Conversation (Capsa).

La pestaña “IPEndpoint” del gráfico D.101 permite obtener información de los nodos con los que estableció comunicación el host seleccionado, además proporciona datos de bytes y paquetes enviados y recibidos.

Además contiene una tabla de conversación IP, TCP, UDP mostrada en la parte inferior del gráfico D.101 que permite analizar de manera más objetiva los valores obtenidos. Estas opciones también se las encuentra en las pestañas de la parte superior.

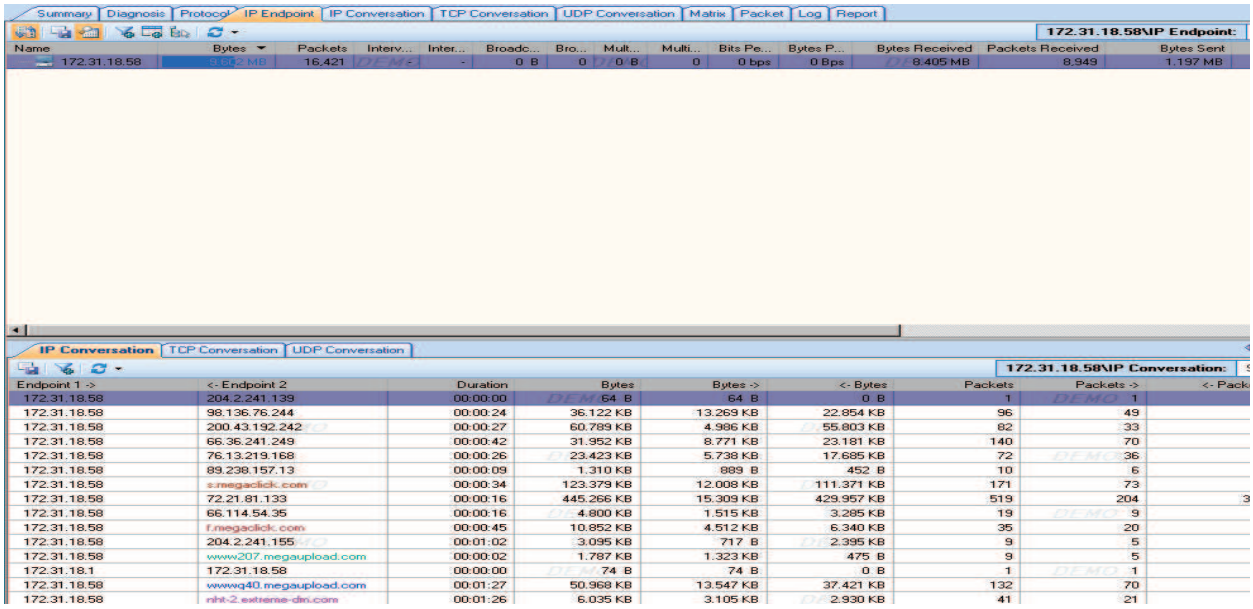


Gráfico D.101 Pestañas IP Endpoint e IP Conversation (Capsa).

La pestaña “IP Conversation” del gráfico D.102 permite mostrar la comunicación y cantidad de datos transferidos del protocolo IP para el host seleccionado y los nodos o servidores remotos con los cuales se conectó durante un periodo de tiempo.

Además contiene una tabla de conversación TCP, UDP mostrada en la parte inferior del gráfico D.102. Estas opciones también se las encuentra en las pestañas de la parte superior.

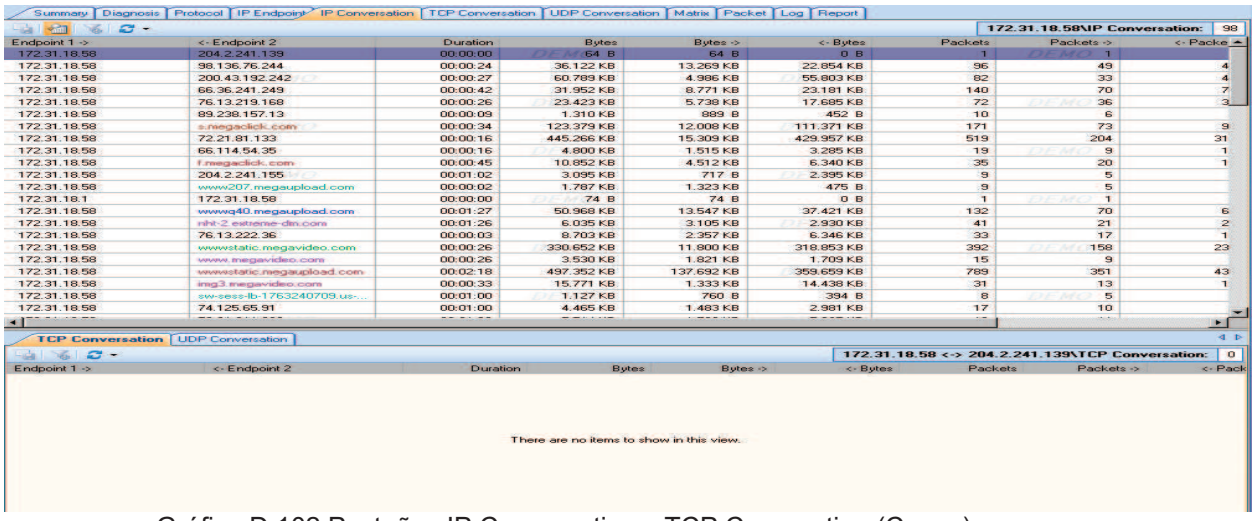


Gráfico D.102 Pestañas IP Conversation y TCP Conersation (Capsa).

La pestaña “TCP Conversation” del gráfico D.103 muestra las conversaciones que usan el protocolo de transporte TCP del host seleccionado con los nodos y servidores con que mantuvo una transferencia de datos. Además contiene información de los bytes y paquetes enviados y recibidos durante el intercambio de información.

En la parte inferior del gráfico D.103 se muestra la pestaña “Packets” que representa la comunicación por puerto, protocolo y su duración mientras existía una transferencia de datos.

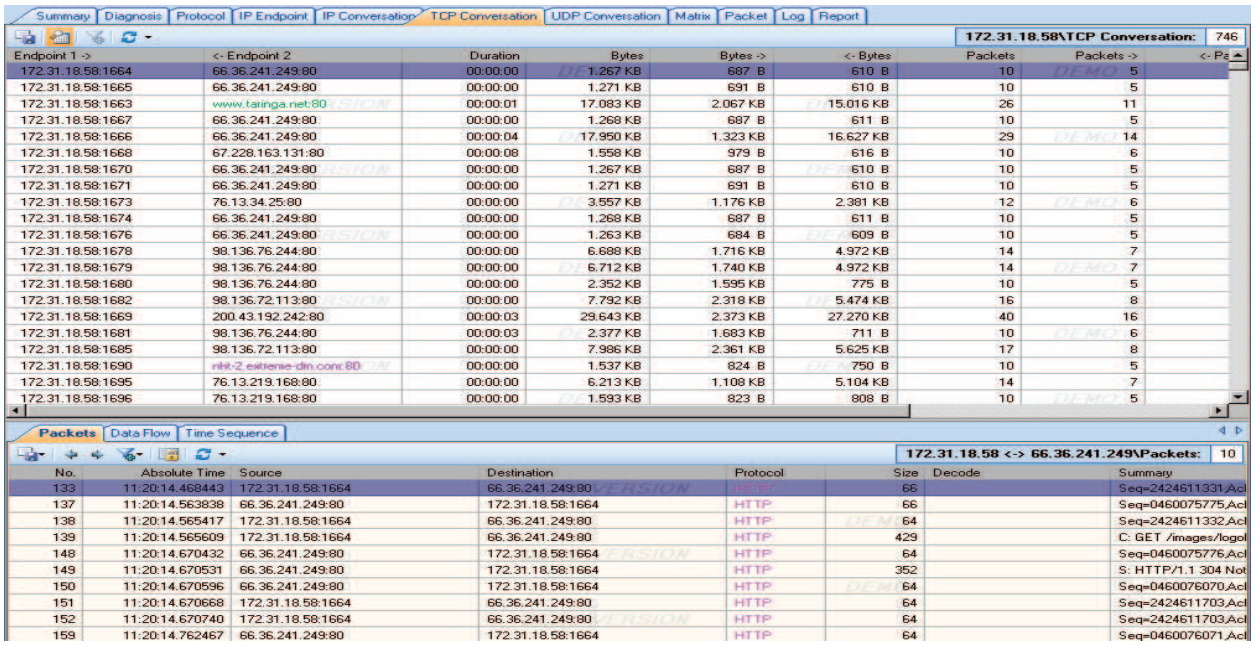


Gráfico D.103 Pestañas TCP Conersation y Packets (Capsa).

La pestaña “Data Flow” del gráfico D.104 perteneciente a la pestaña “TCP Conversation” organiza los paquetes existentes en la conversación en su orden correcto y reconstruye estos en un flujo TCP.

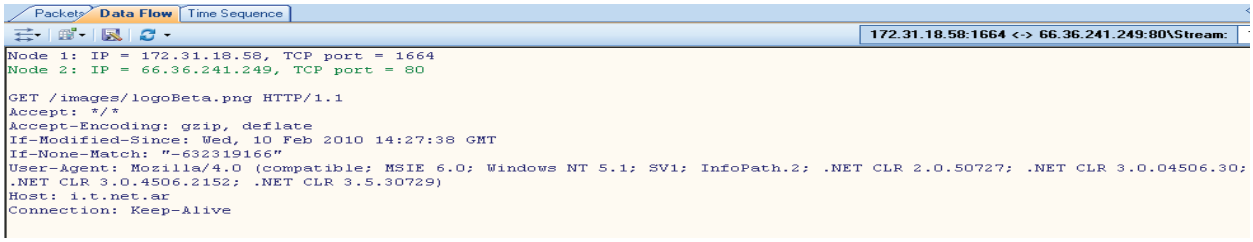


Gráfico D.104 Pestaña Data Flow (Capsa).

La pestaña “Time Sequence” del gráfico D.105 perteneciente a la pestaña “TCP Conversation” muestra las conversaciones TCP con fechas en el orden temporal en que fueron generadas. Además estas flechas se utilizan para comprender la comunicación TCP y analizar diferentes escenarios como la comunicación entre nodos.

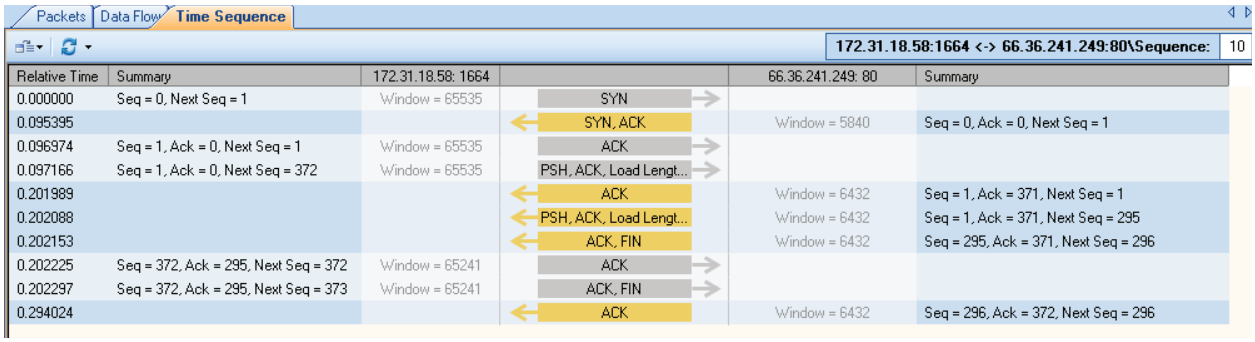


Gráfico D.105 Pestaña Time sequence (Capsa).

La pestaña “UDP Conversation” del gráfico D.106 muestra las conversaciones que usan el protocolo de transporte UDP del host seleccionado, con los nodos y servidores con quien mantuvo comunicación. Además contiene información de los bytes y paquetes enviados y recibidos durante el intercambio de información al igual que la pestaña “TCP Conversation”.

Como se muestra en la parte inferior del gráfico D.106 se incluye la pestaña “Packets” que representa la comunicación por puerto, protocolo y su duración mientras existía una transferencia de datos.

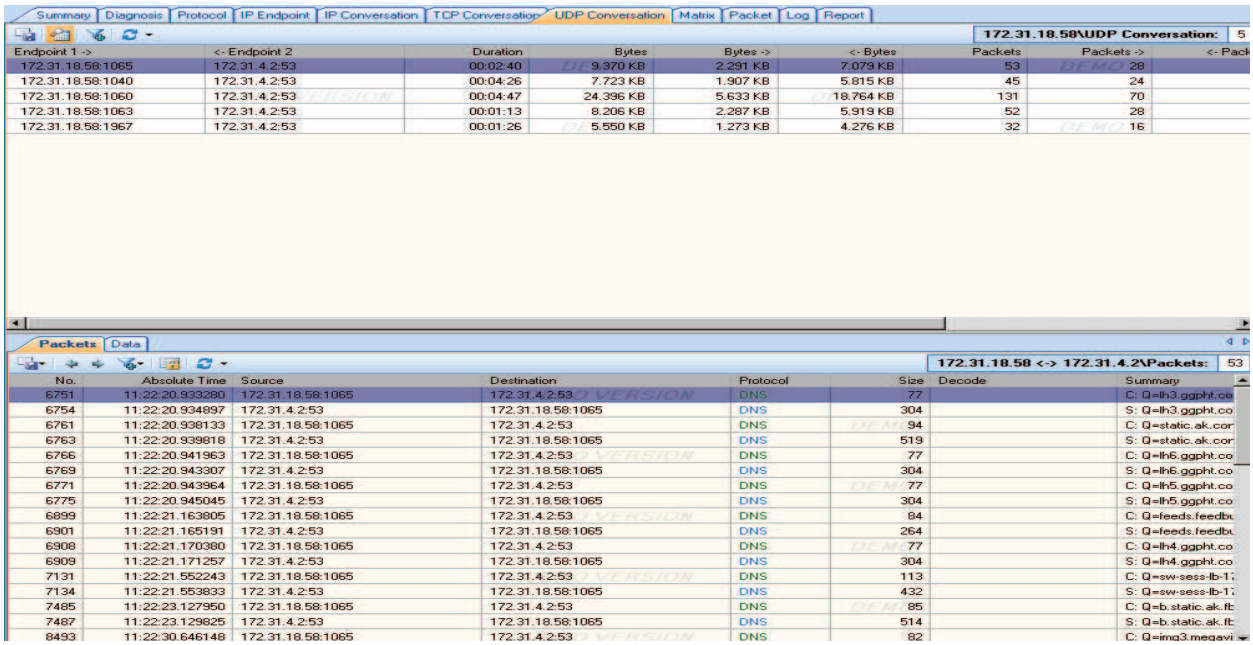


Gráfico D.106 Pestañas UDP Conversation y Packets (Capsa).

La pestaña “Data” del gráfico D.107 perteneciente a “UDP Conversation” reconstruye el flujo de los paquetes monitoreados durante la transferencia de datos en un cuadro de texto. Los flujos de datos de diferentes direcciones se pueden distinguir por el color azul, si la dirección es de 1 a 2, y por color verde, si la dirección es 2 a 1, siendo 1 el host monitoreado y 2 el nodo o servidor remoto.

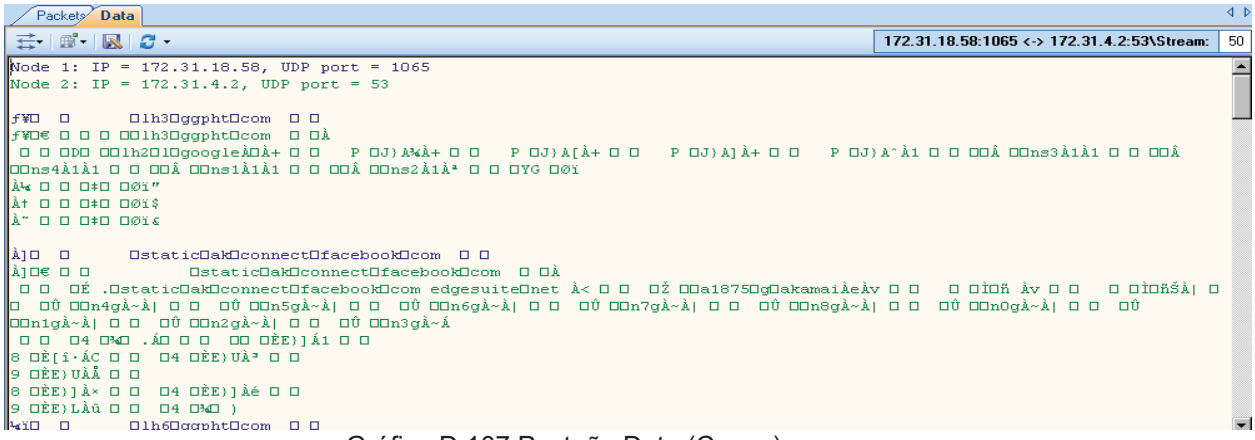


Gráfico D.107 Pestaña Data (Capsa).

La pestaña “Matrix” del gráfico D.108 muestra las diferentes conexiones que realizó el host con los diferentes nodos o servidores remotos y detalla el tráfico de red en una matriz. El espesor de la línea indica el volumen de tráfico entre los nodos con que mantuvo una transferencia de datos.

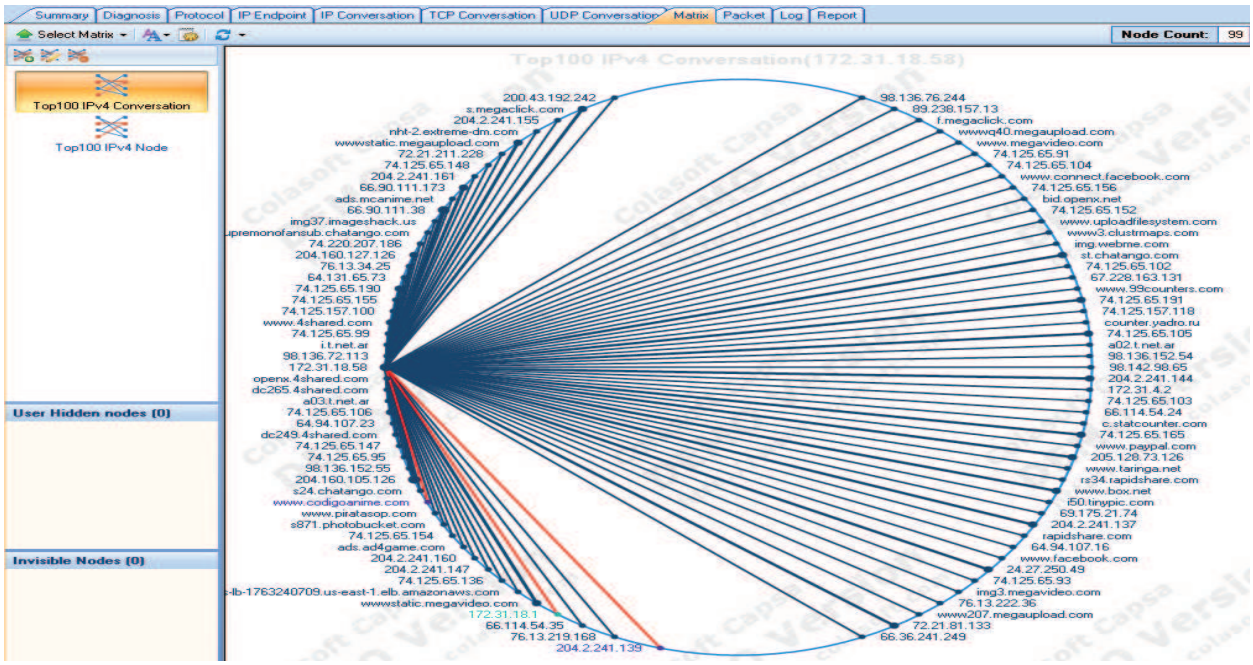


Gráfico D.108 Pestaña Matrix (Capsa).

La pestaña “Packet” del gráfico D.109 muestra los paquetes capturados durante la transferencia de datos y despliega información sobre los valores de los diferentes campos que contiene el paquete que se ha seleccionado. Al elegir cualquier campo del paquete se despliega la información en Hex ASCII de los valores que este contiene.

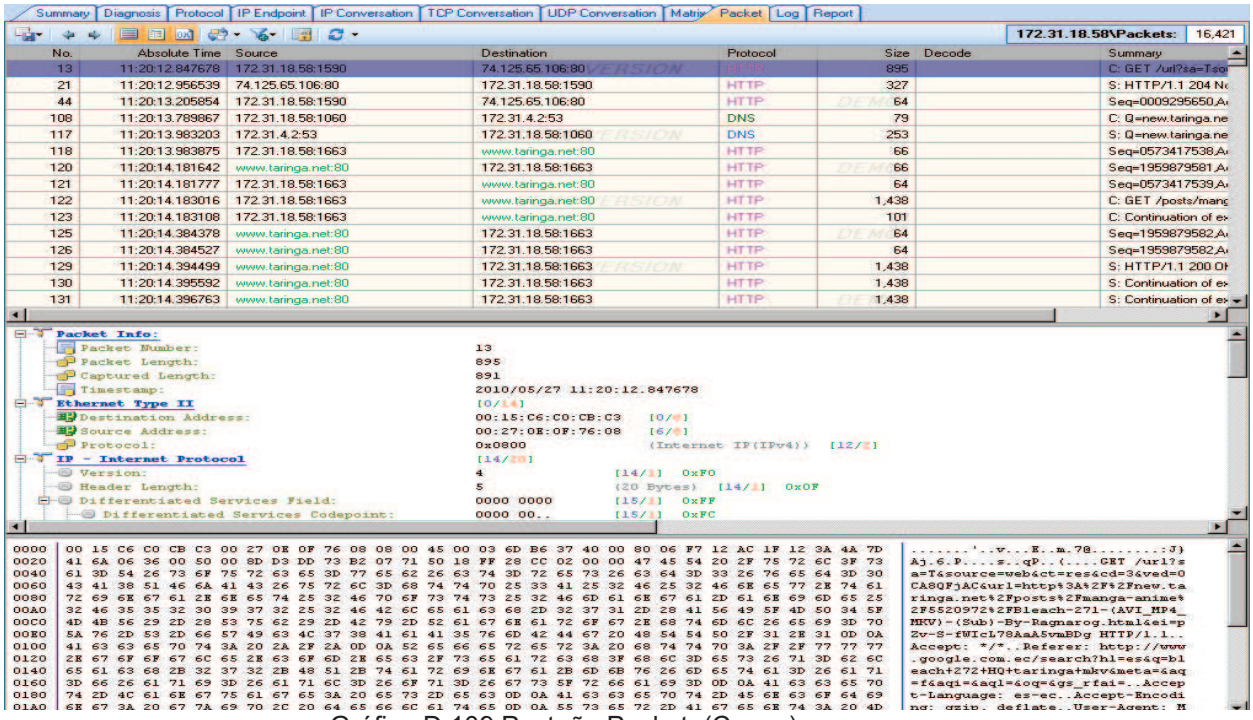


Gráfico D.109 Pestaña Packet (Capsa).

La pestaña “Log” del gráfico D.110 permite mostrar un registro de actividad del host seleccionado en un intervalo de tiempo. Detalla la dirección destino, origen así como el tiempo del evento y el detalle de las operaciones involucradas durante la transferencia de datos.

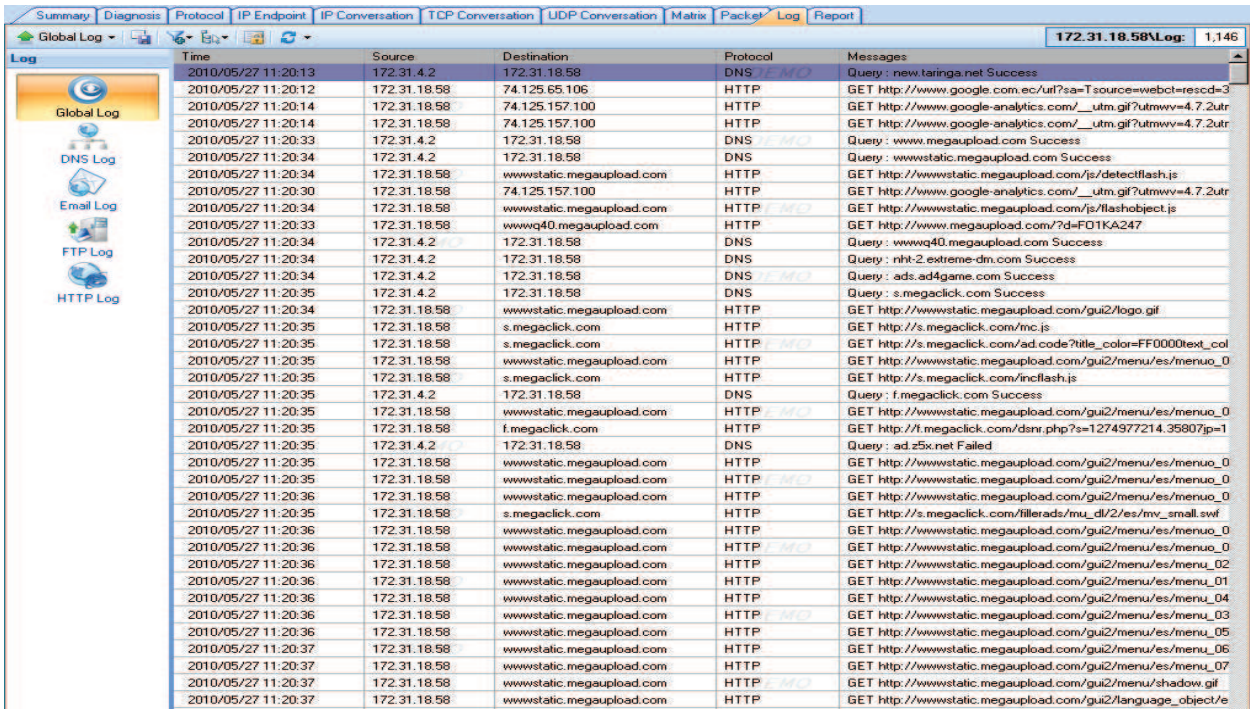


Gráfico D.110 Pestaña Log (Capsa).

La última pestaña “Report” del gráfico D.111 despliega reportes generados durante el monitoreo del host.

Estos reportes son:

- Resumen Estadístico.

- Diagnóstico Estadístico.

- Estadística de Protocolos.

- Protocolos de capa aplicación (10 más usados).

Gráfico D.111 Pestaña Report (Capsa).

La opción de “Resumen Estadístico” del gráfico D.112 muestra una tabla con información de fecha y hora de captura de datos así como su duración.

Summary Statistics

Item					Value
Capture Status					
Start Date					2010/05/23
Start Time					09:01:00
Duration					00:00:02
Traffic	Bytes	Packets	Utility	bps	pps
Total	9.602 MB	16,421	0.000%	0 bps	0
Broadcast	0 B	0	0.000%	0 bps	0
Multicast	0 B	0	0.000%	0 bps	0
Receive	8.405 MB	8,949	-	-	-
Sent	1.197 MB	7,472	-	-	-
Flow					Count
IP Conversation					98
TCP Conversation					746
UDP Conversation					5
TCP					Count
TCP SYN Sent					770
TCP SYN Received					0
TCP SYNACK Sent					0
TCP SYNACK Received					1,015
TCP FIN Sent					229
TCP FIN Received					362
TCP Reset Sent					459
TCP Reset Received					16
DNS Analysis					Count
DNS Query					166
DNS Response					0

Gráfico D.112 Cuadro Summary Statistics (Capsa).

La opción “Diagnóstico Estadístico” mostrada en el gráfico D.113 incluye una tabla con toda la información contenida en la pestaña “Diagnosis” del gráfico D.99.

Diagnosis Statistics

Name	Counts
All Diagnosis	1,033
Application Layer	163
DNS Server Slow Response	3
DNS Host or Domain Does Not Exist	2
DNS Server Error	2
HTTP Suspicious Conversation	1
HTTP Request Page Not Found	5
HTTP Server Returned Error	1
HTTP Server Slow Response	149
Transport Layer	869
TCP Connection Retry	28
TCP Retransmission	32
TCP Slow Response	620
TCP Duplicated Acknowledgement	189
Network Layer	1
ICMP Destination Unreachable	1

Gráfico D.113 Cuadro Diagnosis Statistics (Capsa).

La opción “Estadística de Protocolos” del gráfico D.114 muestra una tabla con toda la información contenida en la pestaña “Protocols” del gráfico D.100.

Protocols Statistics

Name	Percentage		Bytes	Packets
Ethernet II		66.852%	9.602 MB	16,421
IP		66.852%	9.602 MB	16,421
TCP		66.476%	9.548 MB	16,107
HTTP		66.381%	9.534 MB	16,064
Other		0.000%	66 B	1
PPTP		0.000%	66 B	1
HTTPS		0.094%	13.789 KB	41
UDP		0.376%	55.245 KB	313
DNS		0.376%	55.245 KB	313
Query		0.091%	13.392 KB	166
Response		0.282%	41.420 KB	143
Error		0.003%	444 B	4
ICMP		0.000%	74 B	1
Dest Unreach		0.000%	74 B	1

Gráfico D.114 Cuadro Protocols Statistic (Capsa).

La opción “Top 10 Application Protocols” mostrada en el gráfico D.115 incluye los 10 protocolos aplicación, las 10 direcciones físicas, las 10 direcciones IP, 10 direcciones IP locales, y las 10 direcciones IP remotas más usadas por el hosts seleccionado.

Top 10 Application Protocols

Name	Percentage		Bytes	Packets
HTTP		66.381%	9.534 MB	16,064
DNS		0.376%	55.245 KB	313
HTTPS		0.094%	13.789 KB	41
TCP - Other		0.000%	66 B	1
PPTP		0.000%	66 B	1

Gráfico D.115 Cuadro Top 10 Application Protocols (Capsa).

ANEXO E

IEEE Std 830-1998
(Revision of
IEEE Std 830-1993)

IEEE Recommended Practice for Software Requirements Specifications

Sponsor

Software Engineering Standards Committee of the IEEE Computer Society

Approved 25 June 1998

IEEE-SA Standards Board

Abstract: The content and qualities of a good software requirements specification (SRS) are described and several sample SRS outlines are presented. This recommended practice is aimed at specifying requirements of software to be developed but also can be applied to assist in the selection of in-house and commercial software products. Guidelines for compliance with IEEE/EIA 12207.1-1997 are also provided.
Keywords: contract, customer, prototyping, software requirements specification, supplier, system requirements specifications

---

The Institute of Electrical and Electronics Engineers, Inc.
345 East 47th Street, New York, NY 10017-2394, USA

Print: ISBN 0-7381-0332-2, SH94654
PDF: ISBN 0-7381-0448-5, SS94654

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. Members of the committees serve voluntarily and without compensation. They are not necessarily members of the Institute. The standards developed within IEEE represent a consensus of the broad expertise on the subject within the Institute as well as those activities outside of IEEE that have expressed an interest in participating in the development of the standard.

Use of an IEEE Standard is wholly voluntary. The existence of an IEEE Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEEE Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard. Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

Comments for revision of IEEE Standards are welcome from any interested party, regardless of membership affiliation with IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments.

Interpretations: Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of all concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

Note: Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

Introduction

(This introduction is not a part of IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.)

This recommended practice describes recommended approaches for the specification of software requirements. It is based on a model in which the result of the software requirements specification process is an unambiguous and complete specification document. It should help

a) Software customers to accurately describe what they wish to obtain;
b) Software suppliers to understand exactly what the customer wants;
c) Individuals to accomplish the following goals:
    1) Develop a standard software requirements specification (SRS) outline for their own organizations;
    2) Define the format and content of their specific software requirements specifications;
    3) Develop additional local supporting items such as an SRS quality checklist, or an SRS writer's handbook.

To the customers, suppliers, and other individuals, a good SRS should provide several specific benefits, such as the following:

— *Establish the basis for agreement between the customers and the suppliers on what the software product is to do*. The complete description of the functions to be performed by the software specified in the SRS will assist the potential users to determine if the software specified meets their needs or how the software must be modified to meet their needs.

— *Reduce the development effort*. The preparation of the SRS forces the various concerned groups in the customer's organization to consider rigorously all of the requirements before design begins and reduces later redesign, recoding, and retesting. Careful review of the requirements in the SRS can reveal omissions, misunderstandings, and inconsistencies early in the development cycle when these problems are easier to correct.

— *Provide a basis for estimating costs and schedules*. The description of the product to be developed as given in the SRS is a realistic basis for estimating project costs and can be used to obtain approval for bids or price estimates.

— *Provide a baseline for validation and verification*. Organizations can develop their validation and verification plans much more productively from a good SRS. As a part of the development contract, the SRS provides a baseline against which compliance can be measured.

— *Facilitate transfer.* The SRS makes it easier to transfer the software product to new users or new machines. Customers thus find it easier to transfer the software to other parts of their organization, and suppliers find it easier to transfer it to new customers.

— *Serve as a basis for enhancement*. Because the SRS discusses the product but not the project that developed it, the SRS serves as a basis for later enhancement of the finished product. The SRS may need to be altered, but it does provide a foundation for continued production evaluation.

The readers of this document are referred to Annex B for guidelines for using this recommended practice to meet the requirements of IEEE/EIA 12207.1-1997, IEEE/EIA Guide—Industry Implementation of ISO/IEC 12207: 1995, Standard for Information Technology—Software life cycle processes—Life cycle data.

Participants

This recommended practice was prepared by the Life Cycle Data Harmonization Working Group of the Software Engineering Standards Committee of the IEEE Computer Society. At the time this recommended practice was approved, the working group consisted of the following members:

Leonard L. Tripp, *Chair*

Edward Byrne
Paul R. Croll
Perry DeWeese
Robin Fralick
Marilyn Ginsberg-Finner
John Harauz
Mark Henley
Dennis Lawrence
David Maibor
Ray Milovanovic
James Moore
Timothy Niesen
Dennis Rilling
Terry Rout
Richard Schmidt
Norman F. Schneidewind
David Schultz
Basil Sherlund
Peter Voldner
Ronald Wade

The following persons were on the balloting committee:

Syed Ali
Theodore K. Atchinson
Mikhail Auguston
Robert E. Barry
Leo Beltracchi
H. Ronald Berlack
Richard E. Biehl
Michael A. Blackledge
Sandro Bologna
Juris Borzovs
Kathleen L. Briggs
M. Scott Buck
Michael Caldwell
James E. Cardow
Enrico A. Carrara
Lawrence Catchpole
Keith Chan
Antonio M. Cicu
Theo Clarke
Sylvain Clermont
Rosemary Coleman
Virgil Lee Cooper
W. W. Geoff Cozens
Paul R. Croll
Gregory T. Daich
Geoffrey Darnton
Taz Daughtrey
Bostjan K. Derganc
Perry R. DeWeese
James Do
Evelyn S. Dow
Carl Einar Dragstedt
Sherman Eagles
Christof Ebert
Leo Egan
Richard E. Fairley
John W. Fendrich
Jay Forster
Kirby Fortenberry
Eva Freund
Richard C. Fries
Roger U. Fujii
Adel N. Ghannam
Marilyn Ginsberg-Finner
John Garth Glynn
Julio Gonzalez-Sanz
L. M. Gunther
David A. Gustafson
Jon D. Hagar
John Harauz
Robert T. Harley
Herbert Hecht
William Hefley
Manfred Hein
Mark Heinrich
Mark Henley
Debra Herrmann
John W. Horch
Jerry Huller
Peter L. Hung
George Jackelen
Frank V. Jorgensen
William S. Junk
George X. Kambic
Richard Karcich
Ron S. Kenett
Judith S. Kerner
Robert J. Kierzyk
Dwayne L. Knirk
Shaye Koenig
Thomas M. Kurihara
John B. Lane
J. Dennis Lawrence
Fang Ching Lim
William M. Lively
James J. Longbucco
Dieter Look
John Lord
Stan Magee
David Maibor
Harold Mains
Robert A. Martin
Tomoo Matsubara
Mike McAndrew
Patrick D. McCray
Christopher McMacken
Jerome W. Mersky
Bret Michael
Alan Miller
Celia H. Modell
James W. Moore
Pavol Navrat
Myrna L. Olson
Indradeb P. Pal
Alex Polack
Peter T. Poon
Lawrence S. Przybylski
Kenneth R. Ptack
Annette D. Reilly
Dennis Rilling
Andrew P. Sage
Helmut Sandmayr
Stephen R. Schach
Hans Schaefer
Norman Schneidewind
David J. Schultz
Lisa A. Selmon
Robert W. Shillato
David M. Siefert
Carl A. Singer
James M. Sivak
Richard S. Sky
Nancy M. Smith
Melford E. Smyre
Harry M. Sneed
Alfred R. Sorkowitz
Donald W. Sova
Luca Spotorno
Julia Stesney
Fred J. Strauss
Christine Brown Strysik
Toru Takeshita
Richard H. Thayer
Booker Thomas
Patricia Trellue
Theodore J. Urbanowicz
Glenn D. Venables
Udo Voges
David D. Walden
Dolores Wallace
William M. Walsh
John W. Walz
Camille SWhite-Partain
Scott A. Whitmire
P. A. Wolfgang
Paul R. Work
Natalie C. Yopconka
Janusz Zalewski
Geraldine Zimmerman
Peter F. Zoll

When the IEEE-SA Standards Board approved this recommended practice on 25 June 1998, it had the following membership:

Richard J. Holleman, *Chair* **Donald N. Heirman,** *Vice Chair*
Judith Gorman, *Secretary*

Satish K. Aggarwal
Clyde R. Camp
James T. Carlo
Gary R. Engmann
Harold E. Epstein
Jay Forster*
Thomas F. Garrity
Ruben D. Garzon
James H. Gurney
Jim D. Isaak
Lowell G. Johnson
Robert Kennelly
E. G. "Al" Kiener
Joseph L. Koepfinger*
Stephen R. Lambert
Jim Logothetis
Donald C. Loughry
L. Bruce McClung
Louis-François Pau
Ronald C. Petersen
Gerald H. Peterson
John B. Posey
Gary S. Robinson
Hans E. Weinrich
Donald W. Zipse

*Member Emeritus

Valerie E. Zelenty
IEEE Standards Project Editor

Contents

IEEE Recommended Practice for Software Requirements Specifications

1. Overview

This recommended practice describes recommended approaches for the specification of software requirements. It is divided into five clauses. Clause 1 explains the scope of this recommended practice. Clause 2 lists the references made to other standards. Clause 3 provides definitions of specific terms used. Clause 4 provides background information for writing a good SRS. Clause 5 discusses each of the essential parts of an SRS. This recommended practice also has two annexes, one which provides alternate format templates, and one which provides guidelines for compliance with IEEE/EIA 12207.1-1997.

1.1 Scope

This is a recommended practice for writing software requirements specifications. It describes the content and qualities of a good software requirements specification (SRS) and presents several sample SRS outlines.

This recommended practice is aimed at specifying requirements of software to be developed but also can be applied to assist in the selection of in-house and commercial software products. However, application to already-developed software could be counterproductive.

When software is embedded in some larger system, such as medical equipment, then issues beyond those identified in this recommended practice may have to be addressed.

This recommended practice describes the process of creating a product and the content of the product. The product is an SRS. This recommended practice can be used to create such an SRS directly or can be used as a model for a more specific standard.

This recommended practice does not identify any specific method, nomenclature, or tool for preparing an SRS.

2. References

This recommended practice shall be used in conjunction with the following publications.

ASTM E1340-96, Standard Guide for Rapid Prototyping of Computerized Systems.[1]

IEEE Std 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology.[2]

IEEE Std 730-1998, IEEE Standard for Software Quality Assurance Plans.

IEEE Std 730.1-1995, IEEE Guide for Software Quality Assurance Planning.

IEEE Std 828-1998, IEEE Standard for Software Configuration Management Plans.[3]

IEEE Std 982.1-1988, IEEE Standard Dictionary of Measures to Produce Reliable Software.

IEEE Std 982.2-1988, IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software.

IEEE Std 1002-1987 (Reaff 1992), IEEE Standard Taxonomy for Software Engineering Standards.

IEEE Std 1012-1998, IEEE Standard for Software Verification and Validation.

IEEE Std 1012a-1998, IEEE Standard for Software Verification and Validation: Content Map to IEEE/EIA 12207.1-1997.[4]

IEEE Std 1016-1998, IEEE Recommended Practice for Software Design Descriptions.[5]

IEEE Std 1028-1997, IEEE Standard for Software Reviews.

IEEE Std 1042-1987 (Reaff 1993), IEEE Guide to Software Configuration Management.

IEEE P1058/D2.1, Draft Standard for Software Project Management Plans, dated 5 August 1998.[6]

IEEE Std 1058a-1998, IEEE Standard for Software Project Management Plans: Content Map to IEEE/EIA 12207.1-1997.[7]

IEEE Std 1074-1997, IEEE Standard for Developing Software Life Cycle Processes.

IEEE Std 1233, 1998 Edition, IEEE Guide for Developing System Requirements Specifications.[8]

---

[1]ASTM publications are available from the American Society for Testing and Materials, 100 Barr Harbor Drive, West Conshohocken, PA 19428-2959, USA.

[2]IEEE publications are available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA.

[3]As this standard goes to press, IEEE Std 828-1998; IEEE Std 1012a-1998; IEEE Std 1016-1998; and IEEE Std 1233, 1998 Edition are approved but not yet published. The draft standards are, however, available from the IEEE. Anticipated publication date is Fall 1998. Contact the IEEE Standards Department at 1 (732) 562-3800 for status information.

[4]See Footnote 3.

[5]See Footnote 3.

[6]Upon approval of IEEE P1058 by the IEEE-SA Standards Board, this standard will be integrated with IEEE Std 1058a-1998 and published as IEEE Std 1058, 1998 Edition. Approval is expected 8 December 1998.

[7]As this standard goes to press, IEEE Std 1058a-1998 is approved but not yet published. The draft standard is, however, available from the IEEE. Anticipated publication date is December 1998. Contact the IEEE Standards Department at 1 (732) 562-3800 for status information. See Footnote 6.

[8]See Footnote 3.

3. Definitions

In general the definitions of terms used in this recommended practice conform to the definitions provided in IEEE Std 610.12-1990. The definitions below are key terms as they are used in this recommended practice.

3.1 contract: A legally binding document agreed upon by the customer and supplier. This includes the technical and organizational requirements, cost, and schedule for a product. A contract may also contain informal but useful information such as the commitments or expectations of the parties involved.

3.2 customer: The person, or persons, who pay for the product and usually (but not necessarily) decide the requirements. In the context of this recommended practice the customer and the supplier may be members of the same organization.

3.3 supplier: The person, or persons, who produce a product for a customer. In the context of this recommended practice, the customer and the supplier may be members of the same organization.

3.4 user: The person, or persons, who operate or interact directly with the product. The user(s) and the customer(s) are often not the same person(s).

4. Considerations for producing a good SRS

This clause provides background information that should be considered when writing an SRS. This includes the following:

a) Nature of the SRS;
b) Environment of the SRS;
c) Characteristics of a good SRS;
d) Joint preparation of the SRS;
e) SRS evolution;
f) Prototyping;
g) Embedding design in the SRS;
h) Embedding project requirements in the SRS.

4.1 Nature of the SRS

The SRS is a specification for a particular software product, program, or set of programs that performs certain functions in a specific environment. The SRS may be written by one or more representatives of the supplier, one or more representatives of the customer, or by both. Subclause 4.4 recommends both.

The basic issues that the SRS writer(s) shall address are the following:

a) *Functionality.* What is the software supposed to do?
b) *External interfaces*. How does the software interact with people, the system's hardware, other hardware, and other software?
c) *Performance*. What is the speed, availability, response time, recovery time of various software functions, etc.?
d) *Attributes*. What are the portability, correctness, maintainability, security, etc. considerations?
e) *Design constraints imposed on an implementation*. Are there any required standards in effect, implementation language, policies for database integrity, resource limits, operating environment(s) etc.?

The SRS writer(s) should avoid placing either design or project requirements in the SRS.

For recommended contents of an SRS see Clause 5.

4.2 Environment of the SRS

It is important to consider the part that the SRS plays in the total project plan, which is defined in IEEE Std 610.12-1990. The software may contain essentially all the functionality of the project or it may be part of a larger system. In the latter case typically there will be an SRS that will state the interfaces between the system and its software portion, and will place external performance and functionality requirements upon the software portion. Of course the SRS should then agree with and expand upon these system requirements.

IEEE Std 1074-1997 describes the steps in the software life cycle and the applicable inputs for each step. Other standards, such as those listed in Clause 2, relate to other parts of the software life cycle and so may complement software requirements.

Since the SRS has a specific role to play in the software development process, the SRS writer(s) should be careful not to go beyond the bounds of that role. This means the SRS

a) Should correctly define all of the software requirements. A software requirement may exist because of the nature of the task to be solved or because of a special characteristic of the project.
b) Should not describe any design or implementation details. These should be described in the design stage of the project.
c) Should not impose additional constraints on the software. These are properly specified in other documents such as a software quality assurance plan.

Therefore, a properly written SRS limits the range of valid designs, but does not specify any particular design.

4.3 Characteristics of a good SRS

An SRS should be

a) Correct;
b) Unambiguous;
c) Complete;
d) Consistent;
e) Ranked for importance and/or stability;
f) Verifiable;
g) Modifiable;
h) Traceable.

4.3.1 Correct

An SRS is correct if, and only if, every requirement stated therein is one that the software shall meet.

There is no tool or procedure that ensures correctness. The SRS should be compared with any applicable superior specification, such as a system requirements specification, with other project documentation, and with other applicable standards, to ensure that it agrees. Alternatively the customer or user can determine if the SRS correctly reflects the actual needs. Traceability makes this procedure easier and less prone to error (see 4.3.8).

4.3.2 Unambiguous

An SRS is unambiguous if, and only if, every requirement stated therein has only one interpretation. As a minimum, this requires that each characteristic of the final product be described using a single unique term.

In cases where a term used in a particular context could have multiple meanings, the term should be included in a glossary where its meaning is made more specific.

An SRS is an important part of the requirements process of the software life cycle and is used in design, implementation, project monitoring, verification and validation, and in training as described in IEEE Std 1074-1997. The SRS should be unambiguous both to those who create it and to those who use it. However, these groups often do not have the same background and therefore do not tend to describe software requirements the same way. Representations that improve the requirements specification for the developer may be counterproductive in that they diminish understanding to the user and vice versa.

Subclauses 4.3.2.1 through 4.3.2.3 recommend how to avoid ambiguity.

4.3.2.1 Natural language pitfalls

Requirements are often written in natural language (e.g., English). Natural language is inherently ambiguous. A natural language SRS should be reviewed by an independent party to identify ambiguous use of language so that it can be corrected.

4.3.2.2 Requirements specification languages

One way to avoid the ambiguity inherent in natural language is to write the SRS in a particular requirements specification language. Its language processors automatically detect many lexical, syntactic, and semantic errors.

One disadvantage in the use of such languages is the length of time required to learn them. Also, many nontechnical users find them unintelligible. Moreover, these languages tend to be better at expressing certain types of requirements and addressing certain types of systems. Thus, they may influence the requirements in subtle ways.

4.3.2.3 Representation tools

In general, requirements methods and languages and the tools that support them fall into three general categories—object, process, and behavioral. Object-oriented approaches organize the requirements in terms of real-world objects, their attributes, and the services performed by those objects. Process-based approaches organize the requirements into hierarchies of functions that communicate via data flows. Behavioral approaches describe external behavior of the system in terms of some abstract notion (such as predicate calculus), mathematical functions, or state machines.

The degree to which such tools and methods may be useful in preparing an SRS depends upon the size and complexity of the program. No attempt is made here to describe or endorse any particular tool.

When using any of these approaches it is best to retain the natural language descriptions. That way, customers unfamiliar with the notations can still understand the SRS.

4.3.3 Complete

An SRS is complete if, and only if, it includes the following elements:

a) All significant requirements, whether relating to functionality, performance, design constraints, attributes, or external interfaces. In particular any external requirements imposed by a system specification should be acknowledged and treated.

b) Definition of the responses of the software to all realizable classes of input data in all realizable classes of situations. Note that it is important to specify the responses to both valid and invalid input values.

c) Full labels and references to all figures, tables, and diagrams in the SRS and definition of all terms and units of measure.

4.3.3.1 Use of TBDs

Any SRS that uses the phrase "to be determined" (TBD) is not a complete SRS. The TBD is, however, occasionally necessary and should be accompanied by

a) A description of the conditions causing the TBD (e.g., why an answer is not known) so that the situation can be resolved;

b) A description of what must be done to eliminate the TBD, who is responsible for its elimination, and by when it must be eliminated.

4.3.4 Consistent

Consistency refers to internal consistency. If an SRS does not agree with some higher-level document, such as a system requirements specification, then it is not correct (see 4.3.1).

4.3.4.1 Internal consistency

An SRS is internally consistent if, and only if, no subset of individual requirements described in it conflict. The three types of likely conflicts in an SRS are as follows:

a) The specified characteristics of real-world objects may conflict. For example,
   1) The format of an output report may be described in one requirement as tabular but in another as textual.
   2) One requirement may state that all lights shall be green while another may state that all lights shall be blue.

b) There may be logical or temporal conflict between two specified actions. For example,
   1) One requirement may specify that the program will add two inputs and another may specify that the program will multiply them.
   2) One requirement may state that "A" must always follow "B," while another may require that "A and B" occur simultaneously.

c) Two or more requirements may describe the same real-world object but use different terms for that object. For example, a program's request for a user input may be called a "prompt" in one requirement and a "cue" in another. The use of standard terminology and definitions promotes consistency.

4.3.5 Ranked for importance and/or stability

An SRS is ranked for importance and/or stability if each requirement in it has an identifier to indicate either the importance or stability of that particular requirement.

Typically, all of the requirements that relate to a software product are not equally important. Some requirements may be essential, especially for life-critical applications, while others may be desirable.

Each requirement in the SRS should be identified to make these differences clear and explicit. Identifying the requirements in the following manner helps:

a) Have customers give more careful consideration to each requirement, which often clarifies any hidden assumptions they may have.
b) Have developers make correct design decisions and devote appropriate levels of effort to the different parts of the software product.

4.3.5.1 Degree of stability

One method of identifying requirements uses the dimension of stability. Stability can be expressed in terms of the number of expected changes to any requirement based on experience or knowledge of forthcoming events that affect the organization, functions, and people supported by the software system.

4.3.5.2 Degree of necessity

Another way to rank requirements is to distinguish classes of requirements as essential, conditional, and optional.

a) *Essential*. Implies that the software will not be acceptable unless these requirements are provided in an agreed manner.
b) *Conditional*. Implies that these are requirements that would enhance the software product, but would not make it unacceptable if they are absent.
c) *Optional*. Implies a class of functions that may or may not be worthwhile. This gives the supplier the opportunity to propose something that exceeds the SRS.

4.3.6 Verifiable

An SRS is verifiable if, and only if, every requirement stated therein is verifiable. A requirement is verifiable if, and only if, there exists some finite cost-effective process with which a person or machine can check that the software product meets the requirement. In general any ambiguous requirement is not verifiable.

Nonverifiable requirements include statements such as “works well,” “good human interface,” and “shall usually happen.” These requirements cannot be verified because it is impossible to define the terms “good,” “well,” or “usually.” The statement that “the program shall never enter an infinite loop” is nonverifiable because the testing of this quality is theoretically impossible.

An example of a verifiable statement is

> *Output of the program shall be produced within 20 s of event × 60% of the time; and shall be produced within 30 s of event × 100% of the time.*

This statement can be verified because it uses concrete terms and measurable quantities.

If a method cannot be devised to determine whether the software meets a particular requirement, then that requirement should be removed or revised.

4.3.7 Modifiable

An SRS is modifiable if, and only if, its structure and style are such that any changes to the requirements can be made easily, completely, and consistently while retaining the structure and style. Modifiability generally requires an SRS to

a) Have a coherent and easy-to-use organization with a table of contents, an index, and explicit cross-referencing;
b) Not be redundant (i.e., the same requirement should not appear in more than one place in the SRS);
c) Express each requirement separately, rather than intermixed with other requirements.

Redundancy itself is not an error, but it can easily lead to errors. Redundancy can occasionally help to make an SRS more readable, but a problem can arise when the redundant document is updated. For instance, a requirement may be altered in only one of the places where it appears. The SRS then becomes inconsistent. Whenever redundancy is necessary, the SRS should include explicit cross-references to make it modifiable.

4.3.8 Traceable

An SRS is traceable if the origin of each of its requirements is clear and if it facilitates the referencing of each requirement in future development or enhancement documentation. The following two types of traceability are recommended:

a) *Backward traceability (i.e., to previous stages of development)*. This depends upon each requirement explicitly referencing its source in earlier documents.
b) *Forward traceability (i.e., to all documents spawned by the SRS)*. This depends upon each requirement in the SRS having a unique name or reference number.

The forward traceability of the SRS is especially important when the software product enters the operation and maintenance phase. As code and design documents are modified, it is essential to be able to ascertain the complete set of requirements that may be affected by those modifications.

4.4 Joint preparation of the SRS

The software development process should begin with supplier and customer agreement on what the completed software must do. This agreement, in the form of an SRS, should be jointly prepared. This is important because usually neither the customer nor the supplier is qualified to write a good SRS alone.

a) Customers usually do not understand the software design and development process well enough to write a usable SRS.
b) Suppliers usually do not understand the customer's problem and field of endeavor well enough to specify requirements for a satisfactory system.

Therefore, the customer and the supplier should work together to produce a well-written and completely understood SRS.

A special situation exists when a system and its software are both being defined concurrently. Then the functionality, interfaces, performance, and other attributes and constraints of the software are not predefined, but rather are jointly defined and subject to negotiation and change. This makes it more difficult, but no less important, to meet the characteristics stated in 4.3. In particular, an SRS that does not comply with the requirements of its parent system specification is incorrect.

This recommended practice does not specifically discuss style, language usage, or techniques of good writing. It is quite important, however, that an SRS be well written. General technical writing books can be used for guidance.

4.5 SRS evolution

The SRS may need to evolve as the development of the software product progresses. It may be impossible to specify some details at the time the project is initiated (e.g., it may be impossible to define all of the screen formats for an interactive program during the requirements phase). Additional changes may ensue as deficiencies, shortcomings, and inaccuracies are discovered in the SRS.

Two major considerations in this process are the following:

a) Requirements should be specified as completely and thoroughly as is known at the time, even if evolutionary revisions can be foreseen as inevitable. The fact that they are incomplete should be noted.

b) A formal change process should be initiated to identify, control, track, and report projected changes. Approved changes in requirements should be incorporated in the SRS in such a way as to

   1) Provide an accurate and complete audit trail of changes;
   2) Permit the review of current and superseded portions of the SRS.

4.6 Prototyping

Prototyping is used frequently during the requirements portion of a project. Many tools exist that allow a prototype, exhibiting some characteristics of a system, to be created very quickly and easily. See also ASTM E1340-96.

Prototypes are useful for the following reasons:

a) The customer may be more likely to view the prototype and react to it than to read the SRS and react to it. Thus, the prototype provides quick feedback.

b) The prototype displays unanticipated aspects of the systems behavior. Thus, it produces not only answers but also new questions. This helps reach closure on the SRS.

c) An SRS based on a prototype tends to undergo less change during development, thus shortening development time.

A prototype should be used as a way to elicit software requirements. Some characteristics such as screen or report formats can be extracted directly from the prototype. Other requirements can be inferred by running experiments with the prototype.

4.7 Embedding design in the SRS

A requirement specifies an externally visible function or attribute of a system. A design describes a particular subcomponent of a system and/or its interfaces with other subcomponents. The SRS writer(s) should clearly distinguish between identifying required design constraints and projecting a specific design. Note that every requirement in the SRS limits design alternatives. This does not mean, though, that every requirement is design.

The SRS should specify what functions are to be performed on what data to produce what results at what location for whom. The SRS should focus on the services to be performed. The SRS should not normally specify design items such as the following:

a) Partitioning the software into modules;
b) Allocating functions to the modules;
c) Describing the flow of information or control between modules;
d) Choosing data structures.

4.7.1 Necessary design requirements

In special cases some requirements may severely restrict the design. For example, security or safety requirements may reflect directly into design such as the need to

a) Keep certain functions in separate modules;
b) Permit only limited communication between some areas of the program;
c) Check data integrity for critical variables.

Examples of valid design constraints are physical requirements, performance requirements, software development standards, and software quality assurance standards.

Therefore, the requirements should be stated from a purely external viewpoint. When using models to illustrate the requirements, remember that the model only indicates the external behavior, and does not specify a design.

4.8 Embedding project requirements in the SRS

The SRS should address the software product, not the process of producing the software product.

Project requirements represent an understanding between the customer and the supplier about contractual matters pertaining to production of software and thus should not be included in the SRS. These normally include items such as

a) Cost;
b) Delivery schedules;
c) Reporting procedures;
d) Software development methods;
e) Quality assurance;
f) Validation and verification criteria;
g) Acceptance procedures.

Project requirements are specified in other documents, typically in a software development plan, a software quality assurance plan, or a statement of work.

5. The parts of an SRS

This clause discusses each of the essential parts of the SRS. These parts are arranged in Figure 1 in an outline that can serve as an example for writing an SRS.

While an SRS does not have to follow this outline or use the names given here for its parts, a good SRS should include all the information discussed here.

Table of Contents

Figure 1—Prototype SRS outline

5.1 Introduction (Section 1 of the SRS)

The introduction of the SRS should provide an overview of the entire SRS. It should contain the following subsections:

a) Purpose;
b) Scope;
c) Definitions, acronyms, and abbreviations;
d) References;
e) Overview.

5.1.1 Purpose (1.1 of the SRS)

This subsection should

a) Delineate the purpose of the SRS;
b) Specify the intended audience for the SRS.

5.1.2 Scope (1.2 of the SRS)

This subsection should

a) Identify the software product(s) to be produced by name (e.g., Host DBMS, Report Generator, etc.);
b) Explain what the software product(s) will, and, if necessary, will not do;
c) Describe the application of the software being specified, including relevant benefits, objectives, and goals;
d) Be consistent with similar statements in higher-level specifications (e.g., the system requirements specification), if they exist.

5.1.3 Definitions, acronyms, and abbreviations (1.3 of the SRS)

This subsection should provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS. This information may be provided by reference to one or more appendixes in the SRS or by reference to other documents.

5.1.4 References (1.4 of the SRS)

This subsection should

a) Provide a complete list of all documents referenced elsewhere in the SRS;
b) Identify each document by title, report number (if applicable), date, and publishing organization;
c) Specify the sources from which the references can be obtained.

This information may be provided by reference to an appendix or to another document.

5.1.5 Overview (1.5 of the SRS)

This subsection should

a) Describe what the rest of the SRS contains;
b) Explain how the SRS is organized.

5.2 Overall description (Section 2 of the SRS)

This section of the SRS should describe the general factors that affect the product and its requirements. This section does not state specific requirements. Instead, it provides a background for those requirements, which are defined in detail in Section 3 of the SRS, and makes them easier to understand.

This section usually consists of six subsections, as follows:

a) Product perspective;
b) Product functions;
c) User characteristics;
d) Constraints;
e) Assumptions and dependencies;
f) Apportioning of requirements.

5.2.1 Product perspective (2.1 of the SRS)

This subsection of the SRS should put the product into perspective with other related products. If the product is independent and totally self-contained, it should be so stated here. If the SRS defines a product that is a component of a larger system, as frequently occurs, then this subsection should relate the requirements of that larger system to functionality of the software and should identify interfaces between that system and the software.

A block diagram showing the major components of the larger system, interconnections, and external interfaces can be helpful.

This subsection should also describe how the software operates inside various constraints. For example, these constraints could include

a) System interfaces;
b) User interfaces;
c) Hardware interfaces;
d) Software interfaces;
e) Communications interfaces;
f) Memory;
g) Operations;
h) Site adaptation requirements.

5.2.1.1 System interfaces

This should list each system interface and identify the functionality of the software to accomplish the system requirement and the interface description to match the system.

5.2.1.2 User interfaces

This should specify the following:

a) *The logical characteristics of each interface between the software product and its users*. This includes those configuration characteristics (e.g., required screen formats, page or window layouts, content of any reports or menus, or availability of programmable function keys) necessary to accomplish the software requirements.

b) *All the aspects of optimizing the interface with the person who must use the system*. This may simply comprise a list of do's and don'ts on how the system will appear to the user. One example may be a requirement for the option of long or short error messages. Like all others, these requirements should be verifiable, e.g., "a clerk typist grade 4 can do function *X* in *Z* min after 1 h of training" rather than "a typist can do function *X*." (This may also be specified in the Software System Attributes under a section titled Ease of Use.)

5.2.1.3 Hardware interfaces

This should specify the logical characteristics of each interface between the software product and the hardware components of the system. This includes configuration characteristics (number of ports, instruction sets, etc.). It also covers such matters as what devices are to be supported, how they are to be supported, and protocols. For example, terminal support may specify full-screen support as opposed to line-by-line support.

5.2.1.4 Software interfaces

This should specify the use of other required software products (e.g., a data management system, an operating system, or a mathematical package), and interfaces with other application systems (e.g., the linkage between an accounts receivable system and a general ledger system). For each required software product, the following should be provided:

— Name;
— Mnemonic;
— Specification number;
— Version number;
— Source.

For each interface, the following should be provided:

— Discussion of the purpose of the interfacing software as related to this software product.
— Definition of the interface in terms of message content and format. It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.

5.2.1.5 Communications interfaces

This should specify the various interfaces to communications such as local network protocols, etc.

5.2.1.6 Memory constraints

This should specify any applicable characteristics and limits on primary and secondary memory.

5.2.1.7 Operations

This should specify the normal and special operations required by the user such as

a) The various modes of operations in the user organization (e.g., user-initiated operations);
b) Periods of interactive operations and periods of unattended operations;
c) Data processing support functions;
d) Backup and recovery operations.

NOTE—This is sometimes specified as part of the User Interfaces section.

5.2.1.8 Site adaptation requirements

This should

a) Define the requirements for any data or initialization sequences that are specific to a given site, mission, or operational mode (e.g., grid values, safety limits, etc.);
b) Specify the site or mission-related features that should be modified to adapt the software to a particular installation.

5.2.2 Product functions (2.2 of the SRS)

This subsection of the SRS should provide a summary of the major functions that the software will perform. For example, an SRS for an accounting program may use this part to address customer account maintenance, customer statement, and invoice preparation without mentioning the vast amount of detail that each of those functions requires.

Sometimes the function summary that is necessary for this part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product. Note that for the sake of clarity

a) The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the first time.
b) Textual or graphical methods can be used to show the different functions and their relationships. Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables.

5.2.3 User characteristics (2.3 of the SRS)

This subsection of the SRS should describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise. It should not be used to state specific requirements, but rather should provide the reasons why certain specific requirements are later specified in Section 3 of the SRS.

5.2.4 Constraints (2.4 of the SRS)

This subsection of the SRS should provide a general description of any other items that will limit the developer's options. These include

a) Regulatory policies;
b) Hardware limitations (e.g., signal timing requirements);
c) Interfaces to other applications;
d) Parallel operation;
e) Audit functions;
f) Control functions;
g) Higher-order language requirements;
h) Signal handshake protocols (e.g., XON-XOFF, ACK-NACK);
i) Reliability requirements;
j) Criticality of the application;
k) Safety and security considerations.

5.2.5 Assumptions and dependencies (2.5 of the SRS)

This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption may be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.

5.2.6 Apportioning of requirements (2.6 of the SRS)

This subsection of the SRS should identify requirements that may be delayed until future versions of the system.

5.3 Specific requirements (Section 3 of the SRS)

This section of the SRS should contain all of the software requirements to a level of detail sufficient to enable designers to design a system to satisfy those requirements, and testers to test that the system satisfies those requirements. Throughout this section, every stated requirement should be externally perceivable by users, operators, or other external systems. These requirements should include at a minimum a description of every input (stimulus) into the system, every output (response) from the system, and all functions performed by the system in response to an input or in support of an output. As this is often the largest and most important part of the SRS, the following principles apply:

a) Specific requirements should be stated in conformance with all the characteristics described in 4.3.
b) Specific requirements should be cross-referenced to earlier documents that relate.
c) All requirements should be uniquely identifiable.
d) Careful attention should be given to organizing the requirements to maximize readability.

Before examining specific ways of organizing the requirements it is helpful to understand the various items that comprise requirements as described in 5.3.1 through 5.3.7.

5.3.1 External interfaces

This should be a detailed description of all inputs into and outputs from the software system. It should complement the interface descriptions in 5.2 and should not repeat information there.

It should include both content and format as follows:

a) Name of item;
b) Description of purpose;
c) Source of input or destination of output;
d) Valid range, accuracy, and/or tolerance;
e) Units of measure;
f) Timing;
g) Relationships to other inputs/outputs;
h) Screen formats/organization;
i) Window formats/organization;
j) Data formats;
k) Command formats;
l) End messages.

5.3.2 Functions

Functional requirements should define the fundamental actions that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These are generally listed as “shall” statements starting with “The system shall…”

These include

a) Validity checks on the inputs
b) Exact sequence of operations
c) Responses to abnormal situations, including
    1) Overflow
    2) Communication facilities
    3) Error handling and recovery
d) Effect of parameters
e) Relationship of outputs to inputs, including
    1) Input/output sequences
    2) Formulas for input to output conversion

It may be appropriate to partition the functional requirements into subfunctions or subprocesses. This does not imply that the software design will also be partitioned that way.

5.3.3 Performance requirements

This subsection should specify both the static and the dynamic numerical requirements placed on the software or on human interaction with the software as a whole. Static numerical requirements may include the following:

a) The number of terminals to be supported;
b) The number of simultaneous users to be supported;
c) Amount and type of information to be handled.

Static numerical requirements are sometimes identified under a separate section entitled Capacity.

Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.

All of these requirements should be stated in measurable terms.

For example,

95% of the transactions shall be processed in less than 1 s.

rather than,

An operator shall not have to wait for the transaction to complete.

NOTE—Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.

5.3.4 Logical database requirements

This should specify the logical requirements for any information that is to be placed into a database. This may include the following:

a) Types of information used by various functions;
b) Frequency of use;
c) Accessing capabilities;
d) Data entities and their relationships;
e) Integrity constraints;
f) Data retention requirements.

5.3.5 Design constraints

This should specify design constraints that can be imposed by other standards, hardware limitations, etc.

5.3.5.1 Standards compliance

This subsection should specify the requirements derived from existing standards or regulations. They may include the following:

a) Report format;
b) Data naming;
c) Accounting procedures;
d) Audit tracing.

For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.

5.3.6 Software system attributes

There are a number of attributes of software that can serve as requirements. It is important that required attributes be specified so that their achievement can be objectively verified. Subclauses 5.3.6.1 through 5.3.6.5 provide a partial list of examples.

5.3.6.1 Reliability

This should specify the factors required to establish the required reliability of the software system at time of delivery.

5.3.6.2 Availability

This should specify the factors required to guarantee a defined availability level for the entire system such as checkpoint, recovery, and restart.

5.3.6.3 Security

This should specify the factors that protect the software from accidental or malicious access, use, modification, destruction, or disclosure. Specific requirements in this area could include the need to

a) Utilize certain cryptographical techniques;
b) Keep specific log or history data sets;
c) Assign certain functions to different modules;
d) Restrict communications between some areas of the program;
e) Check data integrity for critical variables.

5.3.6.4 Maintainability

This should specify attributes of software that relate to the ease of maintenance of the software itself. There may be some requirement for certain modularity, interfaces, complexity, etc. Requirements should not be placed here just because they are thought to be good design practices.

5.3.6.5 Portability

This should specify attributes of software that relate to the ease of porting the software to other host machines and/or operating systems. This may include the following:

a) Percentage of components with host-dependent code;
b) Percentage of code that is host dependent;
c) Use of a proven portable language;
d) Use of a particular compiler or language subset;
e) Use of a particular operating system.

5.3.7 Organizing the specific requirements

For anything but trivial systems the detailed requirements tend to be extensive. For this reason, it is recommended that careful consideration be given to organizing these in a manner optimal for understanding. There is no one optimal organization for all systems. Different classes of systems lend themselves to different organizations of requirements in Section 3 of the SRS. Some of these organizations are described in 5.3.7.1 through 5.3.7.7.

5.3.7.1 System mode

Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency. When organizing this section by mode, the outline in A.1 or A.2 should be used. The choice depends on whether interfaces and performance are dependent on mode.

5.3.7.2 User class

Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters. When organizing this section by user class, the outline in A.3 should be used.

5.3.7.3 Objects

Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. When organizing this section by object, the outline in A.4 should be used. Note that sets of objects may share attributes and services. These are grouped together as classes.

5.3.7.4 Feature

A feature is an externally desired service by the system that may require a sequence of inputs to effect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs. When organizing this section by feature, the outline in A.5 should be used.

5.3.7.5 Stimulus

Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc. When organizing this section by stimulus, the outline in A.6 should be used.

5.3.7.6 Response

Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc. The outline in A.6 (with all occurrences of stimulus replaced with response) should be used.

5.3.7.7 Functional hierarchy

When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by either common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data. When organizing this section by functional hierarchy, the outline in A.7 should be used.

5.3.8 Additional comments

Whenever a new SRS is contemplated, more than one of the organizational techniques given in 5.3.7.7 may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification. For example, see A.8 for an organization combining user class and feature. Any additional requirements may be put in a separate section at the end of the SRS.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented

analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.

In any of the outlines given in A.1 through A.8, those sections called "Functional Requirement *i*" may be described in native language (e.g., English), in pseudocode, in a system definition language, or in four subsections titled: Introduction, Inputs, Processing, and Outputs.

5.4 Supporting information

The supporting information makes the SRS easier to use. It includes the following:

a) Table of contents;
b) Index;
c) Appendixes.

5.4.1 Table of contents and index

The table of contents and index are quite important and should follow general compositional practices.

5.4.2 Appendixes

The appendixes are not always considered part of the actual SRS and are not always necessary. They may include

a) Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
b) Supporting or background information that can help the readers of the SRS;
c) A description of the problems to be solved by the software;
d) Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the SRS should explicitly state whether or not the appendixes are to be considered part of the requirements.

Annex A

(informative)

SRS templates

A.1 Template of SRS Section 3 organized by mode: Version 1

3. Specific requirements
3.1 External interface requirements
  3.1.1 User interfaces
  3.1.2 Hardware interfaces
  3.1.3 Software interfaces
  3.1.4 Communications interfaces
3.2 Functional requirements
  3.2.1 Mode 1
    3.2.1.1 Functional requirement 1.1
    .
    .
    .
    3.2.1.*n* Functional requirement 1.*n*
  3.2.2 Mode 2
  .
  .
  .
  3.2.*m* Mode *m*
    3.2.*m*.1 Functional requirement *m*.1
    .
    .
    .
    3.2.*m*.*n* Functional requirement *m*.*n*
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.2 Template of SRS Section 3 organized by mode: Version 2

3. Specific requirements
3.1. Functional requirements
  3.1.1 Mode 1
    3.1.1.1 External interfaces
      3.1.1.1.1 User interfaces
      3.1.1.1.2 Hardware interfaces
      3.1.1.1.3 Software interfaces
      3.1.1.1.4 Communications interfaces
    3.1.1.2 Functional requirements
      3.1.1.2.1 Functional requirement 1
      .
      .
      .

3.1.1.2.*n* Functional requirement *n*
3.1.1.3 Performance
3.1.2 Mode 2
.
.
.
3.1.*m* Mode *m*
3.2 Design constraints
3.3 Software system attributes
3.4 Other requirements

A.3 Template of SRS Section 3 organized by user class

3. Specific requirements
3.1 External interface requirements
3.1.1 User interfaces
3.1.2 Hardware interfaces
3.1.3 Software interfaces
3.1.4 Communications interfaces
3.2 Functional requirements
3.2.1 User class 1
3.2.1.1 Functional requirement 1.1
.
.
.
3.2.1.*n* Functional requirement 1.*n*
3.2.2 User class 2
.
.
.
3.2.*m* User class *m*
3.2.*m*.1 Functional requirement *m*.1
.
.
.
3.2.*m*.*n* Functional requirement *m*.*n*
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.4 Template of SRS Section 3 organized by object

3. Specific requirements
3.1 External interface requirements
3.1.1 User interfaces
3.1.2 Hardware interfaces
3.1.3 Software interfaces
3.1.4 Communications interfaces
3.2 Classes/Objects
3.2.1 Class/Object 1

3.2.1.1 Attributes (direct or inherited)
3.2.1.1.1 Attribute 1
.
.
.
3.2.1.1.*n* Attribute *n*
3.2.1.2 Functions (services, methods, direct or inherited)
3.2.1.2.1 Functional requirement 1.1
.
.
.
3.2.1.2.*m* Functional requirement 1.*m*
3.2.1.3 Messages (communications received or sent)
3.2.2 Class/Object 2
.
.
.
3.2.*p* Class/Object *p*
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.5 Template of SRS Section 3 organized by feature

3. Specific requirements
3.1 External interface requirements
3.1.1 User interfaces
3.1.2 Hardware interfaces
3.1.3 Software interfaces
3.1.4 Communications interfaces
3.2 System features
3.2.1 System Feature 1
3.2.1.1 Introduction/Purpose of feature
3.2.1.2 Stimulus/Response sequence
3.2.1.3 Associated functional requirements
3.2.1.3.1 Functional requirement 1
.
.
.
3.2.1.3.*n* Functional requirement *n*
3.2.2 System feature 2
.
.
.
3.2.*m* System feature *m*
.
.
.
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.6 Template of SRS Section 3 organized by stimulus

3. Specific requirements
3.1 External interface requirements
  3.1.1 User interfaces
  3.1.2 Hardware interfaces
  3.1.3 Software interfaces
  3.1.4 Communications interfaces
3.2 Functional requirements
  3.2.1 Stimulus 1
    3.2.1.1 Functional requirement 1.1
    .
    .
    .
    3.2.1.*n* Functional requirement 1.*n*
  3.2.2 Stimulus 2
  .
  .
  .
  3.2.*m* Stimulus m
    3.2.*m*.1 Functional requirement *m*.1
    .
    .
    .
    3.2.*m*.*n* Functional requirement *m*.*n*
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.7 Template of SRS Section 3 organized by functional hierarchy

3. Specific requirements
3.1 External interface requirements
  3.1.1 User interfaces
  3.1.2 Hardware interfaces
  3.1.3 Software interfaces
  3.1.4 Communications interfaces
3.2 Functional requirements
  3.2.1 Information flows
    3.2.1.1 Data flow diagram 1
      3.2.1.1.1 Data entities
      3.2.1.1.2 Pertinent processes
      3.2.1.1.3 Topology
    3.2.1.2 Data flow diagram 2
      3.2.1.2.1 Data entities
      3.2.1.2.2 Pertinent processes
      3.2.1.2.3 Topology
    .
    .
    .
    3.2.1.*n* Data flow diagram *n*

3.2.1.*n*.1 Data entities
3.2.1.*n*.2 Pertinent processes
3.2.1.*n*.3 Topology

3.2.2 Process descriptions
3.2.2.1 Process 1
3.2.2.1.1 Input data entities
3.2.2.1.2 Algorithm or formula of process
3.2.2.1.3 Affected data entities
3.2.2.2 Process 2
3.2.2.2.1 Input data entities
3.2.2.2.2 Algorithm or formula of process
3.2.2.2.3 Affected data entities
.
.
.
3.2.2.*m* Process *m*
3.2.2.*m*.1 Input data entities
3.2.2.*m*.2 Algorithm or formula of process
3.2.2.*m*.3 Affected data entities

3.2.3 Data construct specifications
3.2.3.1 Construct 1
3.2.3.1.1 Record type
3.2.3.1.2 Constituent fields
3.2.3.2 Construct 2
3.2.3.2.1 Record type
3.2.3.2.2 Constituent fields
.
.
.
3.2.3.*p* Construct *p*
3.2.3.*p*.1 Record type
3.2.3.*p*.2 Constituent fields

3.2.4 Data dictionary
3.2.4.1 Data element 1
3.2.4.1.1 Name
3.2.4.1.2 Representation
3.2.4.1.3 Units/Format
3.2.4.1.4 Precision/Accuracy
3.2.4.1.5 Range
3.2.4.2 Data element 2
3.2.4.2.1 Name
3.2.4.2.2 Representation
3.2.4.2.3 Units/Format
3.2.4.2.4 Precision/Accuracy
3.2.4.2.5 Range
.
.
.
3.2.4.*q* Data element *q*
3.2.4.*q*.1 Name
3.2.4.*q*.2 Representation
3.2.4.*q*.3 Units/Format
3.2.4.*q*.4 Precision/Accuracy
3.2.4.*q*.5 Range

3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

A.8 Template of SRS Section 3 showing multiple organizations

3. Specific requirements
3.1 External interface requirements
  3.1.1 User interfaces
  3.1.2 Hardware interfaces
  3.1.3 Software interfaces
  3.1.4 Communications interfaces
3.2 Functional requirements
  3.2.1 User class 1
    3.2.1.1 Feature 1.1
      3.2.1.1.1 Introduction/Purpose of feature
      3.2.1.1.2 Stimulus/Response sequence
      3.2.1.1.3 Associated functional requirements
    3.2.1.2 Feature 1.2
      3.2.1.2.1 Introduction/Purpose of feature
      3.2.1.2.2 Stimulus/Response sequence
      3.2.1.2.3 Associated functional requirements
    .
    .
    .
    3.2.1.*m* Feature 1.*m*
      3.2.1.*m*.1 Introduction/Purpose of feature
      3.2.1.*m*.2 Stimulus/Response sequence
      3.2.1.*m*.3 Associated functional requirements
  3.2.2 User class 2
  .
  .
  .
  3.2.*n* User class *n*
  .
  .
  .
3.3 Performance requirements
3.4 Design constraints
3.5 Software system attributes
3.6 Other requirements

Annex B

(informative)

Guidelines for compliance with IEEE/EIA 12207.1-1997

B.1 Overview

The Software Engineering Standards Committee (SESC) of the IEEE Computer Society has endorsed the policy of adopting international standards. In 1995, the international standard, ISO/IEC 12207, Information technology—Software life cycle processes, was completed. The standard establishes a common framework for software life cycle processes, with well-defined terminology, that can be referenced by the software industry.

In 1995 the SESC evaluated ISO/IEC 12207 and decided that the standard should be adopted and serve as the basis for life cycle processes within the IEEE Software Engineering Collection. The IEEE adaptation of ISO/IEC 12207 is IEEE/EIA 12207.0-1996. It contains ISO/IEC 12207 and the following additions: improved compliance approach, life cycle process objectives, life cycle data objectives, and errata.

The implementation of ISO/IEC 12207 within the IEEE also includes the following:

— IEEE/EIA 12207.1-1997, IEEE/EIA Guide for Information Technology—Software life cycle processes—Life cycle data;

— IEEE/EIA 12207.2-1997, IEEE/EIA Guide for Information Technology—Software life cycle processes—Implementation considerations; and

— Additions to 11 SESC standards (i.e., IEEE Stds 730, 828, 829, 830, 1012, 1016, 1058, 1062, 1219, 1233, 1362) to define the correlation between the data produced by existing SESC standards and the data produced by the application of IEEE/EIA 12207.1-1997.

NOTE—Although IEEE/EIA 12207.1-1997 is a guide, it also contains provisions for application as a standard with specific compliance requirements. This annex treats 12207.1-1997 as a standard.

B.1.1 Scope and purpose

Both IEEE Std 830-1998 and IEEE/EIA 12207.1-1997 place requirements on a Software Requirements Description Document. The purpose of this annex is to explain the relationship between the two sets of requirements so that users producing documents intended to comply with both standards may do so.

B.2 Correlation

This clause explains the relationship between IEEE Std 830-1998 and IEEE/EIA 12207.0-1996 and IEEE/EIA 12207.1-1997 in the following areas: terminology, process, and life cycle data.

B.2.1 Terminology correlation

Both this recommended practice and IEEE/EIA 12207.0-1996 have similar semantics for the key terms of software, requirements, specification, supplier, developer, and maintainer. This recommended practice uses

the term “customer” where IEEE/EIA 12207.0-1996 uses “acquirer,” and this recommended practice uses “user” where IEEE/EIA 12207.0-1996 uses “operator.”

B.2.2 Process correlation

IEEE/EIA 12207.0-1996 uses a process-oriented approach for describing the definition of a set of requirements for software. This recommended practice uses a product-oriented approach, where the product is a Software Requirements Description (SRD). There are natural process steps, namely the steps to create each portion of the SRD. These may be correlated with the process requirements of IEEE/EIA 12207.0-1996. The difference is that this recommended practice is focused on the development of software requirements whereas IEEE/EIA 12207.0-1996 provides an overall life cycle view and mentions Software Requirements Analysis as part of its Development Process. This recommended practice provides a greater level of detail on what is involved in the preparation of an SRD.

B.2.3 Life cycle data correlation

IEEE/EIA 12207.0-1996 takes the viewpoint that the software requirements are derived from the system requirements. Therefore, it uses the term, “description” rather that “specification” to describe the software requirements. In a system in which software is a component, each requiring its own specification, there would be a System Requirements Specification (SRS) and one or more SRDs. If the term Software Requirements Specification had been used, there would be a confusion between an SRS referring to the system or software requirements. In the case where there is a stand-alone software system, IEEE/EIA 12207.1-1997 states “If the software is a stand-alone system, then this document should be a specification.”

B.3 Content mapping

This clause provides details bearing on a claim that an SRS complying with this recommended practice would also achieve “document compliance” with the SRD described in IEEE/EIA 12207.1-1997. The requirements for document compliance are summarized in a single row of Table 1 of IEEE/EIA 12207.1-1997. That row is reproduced in Table B.1 of this recommended practice.

Table B.1—Summary of requirements for an SRD excerpted from Table 1 of IEEE/EIA 12207.1-1997

Information item	IEEE/EIA 12207.0-1996 Clause	Kind	IEEE/EIA 12207.1-1997 Clause	References
Software Requirements Description	5.1.1.4, 5.3.4.1, 5.3.4.2	Description (See note for 6.22.1 of IEEE/EIA 12207.1-1997.)	6.22	IEEE Std 830-1998; EIA/IEEE J-STD-016, F.2.3, F.2.4; MIL-STD 961D. Also see ISO/IEC 5806, 5807, 6593, 8631, 8790, and 11411 for guidance on use of notations.

The requirements for document compliance are discussed in the following subclauses:

— B.3.1 discusses compliance with the information requirements noted in column 2 of Table B.1 as prescribed by 5.1.1.4, 5.3.4.1, and 5.3.4.2 of IEEE/EIA 12207.0-1996.

— B.3.2 discusses compliance with the generic content guideline (the “kind” of document) noted in column 3 of Table B.1 as a “description”. The generic content guidelines for a “description” appear in 5.1 of IEEE/EIA 12207.1-1997.

— B.3.3 discusses compliance with the specific requirements for a Software Requirements Description noted in column 4 of Table B.1 as prescribed by 6.22 of IEEE/EIA 12207.1-1997.

— B.3.4 discusses compliance with the life cycle data objectives of Annex H of IEEE/EIA 12207.0-1996 as described in 4.2 of IEEE/EIA 12207.1-1997.

B.3.1 Compliance with information requirements of IEEE/EIA 12207.0-1996

The information requirements for an SRD are those prescribed by 5.1.1.4, 5.3.4.1, and 5.3.4.2 of IEEE/EIA 12207.0-1996. The requirements are substantively identical to those considered in B.3.3 of this recommended practice.

B.3.2 Compliance with generic content guidelines of IEEE/EIA 12207.1-1997

According to IEEE/EIA 12207.1-1997, the generic content guideline for an SRD is generally a description, as prescribed by 5.1 of IEEE/EIA 12207.1-1997. A complying description shall achieve the purpose stated in 5.1.1 and include the information listed in 5.1.2 of IEEE/EIA 12207.1-1997.

The purpose of a description is:

> IEEE/EIA 12207.1-1997, subclause 5.1.1: Purpose: Describe a planned or actual function, design, performance, or process.

An SRD complying with this recommended practice would achieve the stated purpose.

Any description or specification complying with IEEE/EIA 12207.1-1997 shall satisfy the generic content requirements provided in 5.1.2 of that standard. Table B.2 of this recommended practice lists the generic content items and, where appropriate, references the clause of this recommended practice that requires the same information.

Table B.2—Coverage of generic description requirements by IEEE Std 830-1998

IEEE/EIA 12207.1-1997 generic content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
a) Date of issue and status	—	Date of issue and status shall be provided.
b) Scope	5.1.1 Scope	—
c) Issuing organization	—	Issuing organization shall be identified.
d) References	5.1.4 References	—
e) Context	5.1.2 Scope	—
f) Notation for description	4.3 Characteristics of a good SRS	—
g) Body	5. The parts of an SRS	—
h) Summary	5.1.1. Overview	—
i) Glossary	5.1.3 Definitions	—
j) Change history	—	Change history for the SRD shall be provided or referenced.

B.3.3 Compliance with specific content requirements of IEEE/EIA 12207.1-1997

The specific content requirements for an SRD in IEEE/EIA 12207.1-1997 are prescribed by 6.22 of IEEE/EIA 12207.1-1997. A compliant SRD shall achieve the purpose stated in 6.22.1 of IEEE/EIA 12207.1-1997.

The purpose of the SRD is:

> IEEE/EIA 12207.1-1997, subclause 6.22.1: Purpose: Specify the requirements for a software item and the methods to be used to ensure that each requirement has been met. Used as the basis for design and qualification testing of a software item.

An SRS complying with this recommended practice and meeting the additional requirements of Table B.3 of this recommended practice would achieve the stated purpose.

An SRD compliant with IEEE/EIA 12207.1-1997 shall satisfy the specific content requirements provided in 6.22.3 and 6.22.4 of that standard. Table B.3 of this recommended practice lists the specific content items and, where appropriate, references the clause of this recommended practice that requires the same information.

An SRD specified according the requirements stated or referenced in Table B.3 of this recommended practice shall be evaluated considering the criteria provided in 5.3.4.2 of IEEE/EIA 12207.0-1996.

Table B.3—Coverage of specific SRD requirements by IEEE Std 830-1998

IEEE/EIA 12207.1-1997 specific content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
a) Generic description information	See Table B.2	—
b) System identification and overview	5.1.1 Scope	—
c) Functionality of the software item including: – Performance requirements – Physical characteristics – Environmental conditions	5.3.2 Functions 5.3.3 Performance requirements	Physical characteristics and environmental conditions should be provided.
d) Requirements for interfaces external to software item	5.3.1 External interfaces	—
e) Qualification requirements	—	The requirements to be used for qualification testing should be provided (or referenced).
f) Safety specifications	5.2.4 Constraints	—
g) Security and privacy specifications	5.3.6.3 Security	—
h) Human-factors engineering requirements	5.2.3 User characteristics 5.2.1.2 User interfaces	—
i) Data definition and database requirements	5.3.4 Logical data base requirements	—
j) Installation and acceptance requirements at operation site	5.2.1.8 Site adaptation requirements	Installation and acceptance requirements at operation site
k) Installation and acceptance requirements at maintenance site	—	Installation and acceptance requirements at maintenance site
l) User documentation requirements	—	User documentation requirements
m) User operation and execution requirements	5.2.1.7 Operations	User execution requirements

Table B.3—Coverage of specific SRD requirements by IEEE Std 830-1998 (continued)

IEEE/EIA 12207.1-1997 specific content	Corresponding clauses of IEEE Std 830-1998	Additions to requirements of IEEE Std 830-1998
n) User maintenance requirements	5.3.6.4 Maintainability	—
o) Software quality characteristics	5.3.6 Software system attributes	—
p) Design and implementation constraints	5.2.4 Constraints	—
q) Computer resource requirements	5.3.3 Performance requirements	Computer resource requirements
r) Packaging requirements	—	Packaging requirements
s) Precedence and criticality of requirements	5.2.6 Apportioning of requirements	—
t) Requirements traceability	4.3.8 Traceable	—
u) Rationale	5.2.5 Assumptions and dependencies	—
Items a) through f) below are from 6.22.4 a) Support the life cycle data objectives of Annex H of IEEE/EIA 12207.0-1996	—	Support the life cycle data objectives of Annex H of IEEE/EIA 12207.0-1996
b) Describe any function using well-defined notation	4.3 Characteristics of a good SRS	—
c) Define no requirements that are in conflict	4.3 Characteristics of a good SRS	—
d) User standard terminology and definitions	5.1.3 Definition	—
e) Define each unique requirement one to prevent inconsistency	4.3 Characteristics of a good SRS	—
f) Uniquely identify each requirement	4.3 Characteristics of a good SRS	—

B.3.4 Compliance with life cycle data objectives

In addition to the content requirements, life cycle data shall be managed in accordance with the objectives provided in Annex H of IEEE/EIA 12207.0-1996.

B.4 Conclusion

The analysis suggests that any SRS complying with this recommended practice and the additions shown in Table B.2 and Table B.3 also complies with the requirements of an SRD in IEEE/EIA 12207.1-1997. In addition, to comply with IEEE/EIA 12207.1-1997, an SRS shall support the life cycle data objectives of Annex H of IEEE/EIA 12207.0-1996.

To order IEEE standards…

Call 1. 800. 678. IEEE (4333) in the US and Canada.

Outside of the US and Canada:
1. 732. 981. 0600

To order by fax:
1. 732. 981. 9667

IEEE business hours: 8 a.m.–4:30 p.m. (EST)

For on-line access to IEEE standards information…

Via the World Wide Web:
http://standards.ieee.org/

Via ftp:
stdsbbs.ieee.org

ISBN 0-7381-0332-2

ANEXO F

ESCUELA POLITECNICA NACIONAL

PLAN DEL PROYECTO DE TITULACIÓN

Propuesto por: **Velasco Saulo** **Ortega Galo**	**Áreas Técnicas del Tema:** **Administración y Gestión de Redes, Evaluación de Redes, Redes de Área Local, TCP/IP, Estadística, Programación.**
Director del Proyecto: **Ing. Xavier Calderón**	**Fecha:** **2009-04-03**

1. Tema o Título del proyecto

Creación de software de análisis estadístico del tráfico de Internet aplicable a una Red de Área Local.

2. Objetivos

1. Objetivo General

Crear un software de análisis estadístico utilizando herramientas de procesamiento matemático del tráfico monitorizado en una Red de Área Local para la toma de decisiones de optimización del ancho de banda de una conexión a Internet.

2. Objetivos Específicos

Estudiar el flujo de datos de entrada y salida que utilice una conexión a Internet como enlace a la Red de Área Local.

Diseñar un software en un lenguaje que soporte análisis estadístico usando una metodología de desarrollo adecuada para la obtención de parámetros, que serán necesarios para la toma de decisiones.

Realizar pruebas en un ambiente LAN con conexión a Internet para la verificación del correcto funcionamiento del software diseñado.

Comparar el software diseñado con otros que presenten servicios similares.

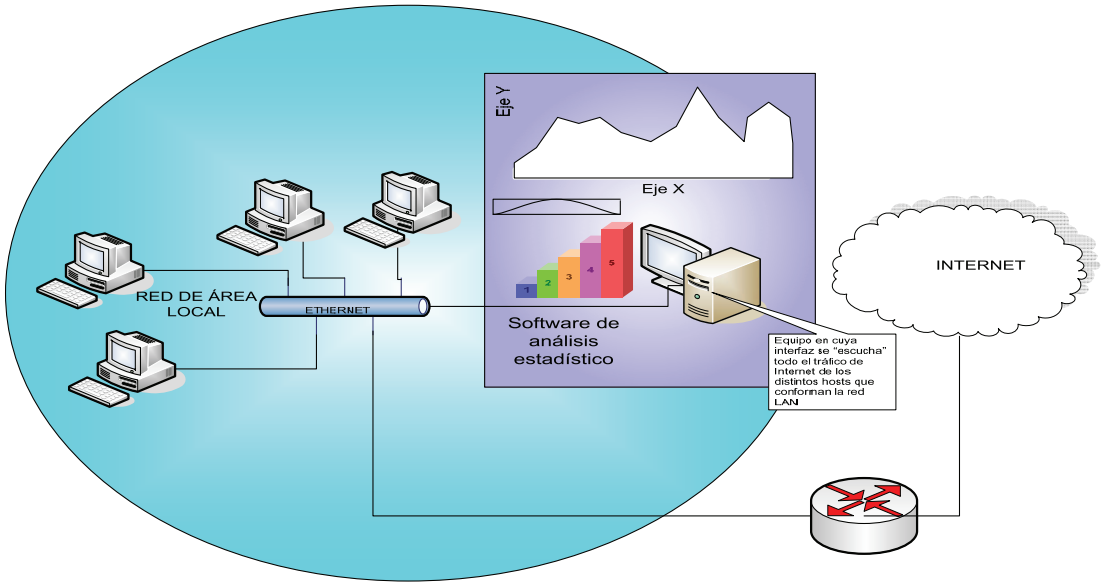
Estimar un costo referencial del desarrollo del proyecto.

3. Alcance

El estudio de tráfico pretende obtener valores cuantitativos sobre el ancho de banda utilizado por los host de una red en cuanto al uso del servicio de internet. Para ello se recabará información de distintas fuentes bibliográficas así como también de Internet.

El software será diseñado bajo una metodología de desarrollo, que permitirá la visualización de la actividad de los host de la red cuando hagan uso del servicio de Internet. Hay que diferenciar dos aspectos importantes de los cuales se hará cargo este software; el primero consiste en la obtención de parámetros estadísticos como son las medidas de tendencia central (Media Aritmética, Mediana, Moda) y medidas de dispersión, partiendo de un conjunto de datos recolectados en un período de tiempo, los cuales serán presentados mediante gráficas estadísticas, además de estos parámetros se incluirá recomendaciones que guíen al administrador de red sobre qué acción tomar frente a situaciones que comprometan el rendimiento y disponibilidad de la conexión a Internet. El software permitirá mostrar los resultados diferenciando hosts, puertos y protocolos de capa aplicación, manteniendo un mecanismo de almacenamiento, para un análisis futuro de los datos recolectados. El segundo punto se refiere a la monitorización en tiempo real de la carga de tráfico para la conexión a Internet la misma que se despliega gráficamente en la interfaz de usuario.

Las pruebas de funcionamiento consistirán en la instalación del software en un host. Este equipo debe poseer una interfaz que permita “escuchar” todas las conexiones que los equipos realicen con servicios a través de internet. Con la recopilación de datos se procederá a realizar los análisis respectivos. Una vez que se hayan tomado acciones para optimizar o limitar el acceso a varios servicios, se pondrá en funcionamiento el software nuevamente para determinar el efecto de estos cambios.



Se realizará la comparación cualitativa del software a implementar con otros que presenten servicios similares mostrando las diferencias entre las opciones existentes.

De la misma manera se calculará una estimación de costos para implementación comercial del software a desarrollar.

4. Justificación del Proyecto

El uso óptimo de los recursos ha sido siempre la principal preocupación para quienes administran una empresa. En la actualidad y con el desarrollo de la redes de comunicaciones este panorama no ha cambiado. Dentro de una Red de Área Local el punto más conflictivo y crítico, tanto por su alto costo como por la dificultad de tener un control directo sobre su utilización, es el servicio de Internet.

El administrador de red tiene la labor de realizar las acciones necesarias para optimizar este recurso. Para ello muchas veces recurre a herramientas de monitorización de ancho de banda para la conexión a Internet. El problema surge cuando se requiere interpretar estos datos para la toma de decisiones, con el objetivo de optimizar el uso del ancho de banda de la conexión de Internet. En estos casos el administrador hace uso de su experiencia y conocimientos, es decir, no tiene parámetros cuantitativos o cualitativos sobre los que basar sus acciones.

En la actualidad existen varias herramientas de análisis de tráfico de una conexión a Internet que muestran algún tipo de tratamiento de datos, éstas hacen un trato de la información de manera superficial ya que solo se supeditan a mostrar de manera gráfica los valores de ancho de banda de tráfico que fueron obtenidos a lo largo del tiempo de muestreo, con el único adicional de un valor promedio y un pico máximo. Se deja al administrador la interpretación de los datos sin ninguna recomendación sobre los resultados obtenidos.

Por otro lado, la mayoría de estos programas muestran los resultados en una interfaz web, sobre algunas distribuciones de Linux y los procedimientos para la instalación de dichos paquetes requieren componentes de software adicionales, que conllevan a confusión y en otros casos a una incorrecta instalación, por lo cual su uso requiere de bastante experiencia en el manejo de este tipo de plataformas y de los lenguajes en los que fueron desarrollados. En el caso de Windows la mayoría de herramientas disponibles tienen un costo monetario relativamente alto. Estas son desarrolladas por empresas privadas. Existen otras herramientas gratuitas pero de alcance limitado o de gran complejidad para los requerimientos de instalación, ya que cada componente adicional requiere un tratamiento independiente.

De esta manera surge la necesidad de diseñar una herramienta, que una vez que realice la adquisición de datos haga un tratamiento estadístico de los mismos. A diferencia de otros programas de análisis de tráfico, este incluirá recomendaciones para el administrador de red, siendo esto una característica novedosa en este tipo de implementaciones. Es de gran ayuda manejar opciones que permitan hacer diferenciaciones, tanto para los distintos hosts que conformen la red como para el uso de servicios o protocolos. Mantener los datos recolectados disponibles para análisis posteriores, permite al administrador de red observar la evolución del uso del ancho de banda, de esta manera podrá verificar la efectividad de las acciones tomadas para la optimización de este recurso.

5. Temas Afines Realizados

1.- Estudio de los factores técnicos y operativos que intervienen en la infraestructura de calidad de servicios en internet / Diego Loor Fonseca y Luis Pichoasamín Morales; dirigido por María Soledad Jiménez Jiménez. - 2001

Se aporta con una nueva herramienta para determinar un análisis más detallado del tráfico de Internet para una Red de Área Local, pudiendo contribuir al estudio de la calidad de servicio y con valores cuantitativos del uso del mismo.

2.- Desarrollo de un sistema de monitoreo para la obtención de la información de red y el gráfico de su topología, basado en la utilización de los protocolos SNMP e ICMP / Carlos Alonso Contreras Gallo; dirigido por Carlos Herrera. – 2006

El proyecto planteado ayudaría a complementar la tesis citada ya que se mostraría de manera más detallada el comportamiento del tráfico de una conexión a Internet por cada host que conforma la red.

3.- Implementación de un software para el monitoreo y obtención de reportes de una red de datos y su aplicación en la auditoría de una red operante / Santiago Javier Oñate Chávez; dirigido por Pablo Hidalgo Lascano. - Quito: 2007

Este proyecto sería un complemento para la realización de una auditoría orientada al uso de los servicios disponibles de Internet para cualquier empresa.

6. Temario

1.- Introducción

Se mostrará brevemente el fundamento teórico en el cual se basará el desarrollo del presente proyecto, mostrando lo más relevante que aportará de forma significativa en el desarrollo del mismo. Esto involucra temas como Estadística Descriptiva, Protocolos de Capa Aplicación y Puertos usados en Internet, API de Programación Orientado a Captura de Paquetes y Análisis de Red e Información sobre la plataforma del desarrollo de software.

2.- Requerimientos

Se presentarán los requerimientos que se tienen para el desarrollo del software.

3.- Desarrollo del software

Se describirá el desarrollo del software utilizando una metodología apropiada, con el lenguaje de programación más conveniente, incluyendo el análisis estadístico para cumplir con los objetivos mencionados anteriormente.

4.- Implementación del prototipo, pruebas de monitoreo y análisis de costos

Se ejecutarán las pruebas de campo con el software implementado. Se describirá detalladamente el funcionamiento del mismo, mostrando los datos obtenidos y comparándolo con un software desarrollado para fines semejantes.

Se realizará un análisis de costos de la implementación del proyecto, para la viabilidad del mismo.

5.- Conclusiones y Recomendaciones

Este capítulo contendrá las conclusiones y recomendaciones a las que se llegue luego de la culminación del proyecto.