Universidad de la República –
Facultad de Ingeniería –
Instituto de Ingeniería Eléctrica

MAPER

Material de Apoyo Pedagógico Enfocado a Redes de datos

Informe Final

Autores
Gastón Arismendi Pereyra
Luís A. da Rosa Fros
Rosana M. Sosa Ferreira

Tutores
Ing. Víctor González Barbone
Ing. Pablo Belzarena

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería Eléctrica. Montevideo, 2008

Tabla de Contenidos

Índice de Figuras

Resumen

El presente informe describe un trabajo realizado como proyecto de grado en Ingeniería Eléctrica. Este proyecto propone el desarrollo de una serie de prácticas de laboratorio para cursos vinculados a redes de datos con enfoque *e-learning*. Para esto se incorpora un simulador gráfico de redes a los objetos de aprendizaje. Se elige un simulador ya existente pero que tiene errores, por lo que la primera etapa de este trabajo aborda las correcciones de dichos errores. El presente informe detalla todas las etapas del proyecto. Se comienza por plantear los objetivos y explicar la planificación realizada. Luego se presenta el desarrollo del proyecto y las herramientas utilizadas. Finalmente se describen los productos, planteándose conclusiones y trabajos a futuro.

Palabras clave: Simulador de redes, Práctica de laboratorio, SCORM, Redes de Datos.

Capítulo 1: Introducción

Actualmente, gran parte de las asignaturas dictadas en la Facultad de Ingeniería de la Universidad de la República (UdelaR) sufren problemas de masividad. En mayor o menor medida, todos los cursos deben ir adaptándose a la nueva realidad. Los docentes buscan mejorar los cursos, utilizando las páginas web de los mismos para brindar mayor y mejor material a los estudiantes. En particular, las asignaturas del área de redes de datos se ven muy afectadas por esta situación debido a que la comprensión de los temas que abarcan requiere realizar prácticas de laboratorio. Dichas prácticas se encuentran limitadas por factores físicos y económicos. En la situación actual, lo que impone la mayor limitante es el laboratorio de *software* del instituto, donde se realizan las ya mencionadas prácticas. Los recursos presentes en dicho laboratorio son, a nuestro entender, escasos y costosos. Escasos porque el laboratorio debe ser compartido por varias asignaturas y tiene pocas máquinas. Costosos por el alto precio de mercado de las máquinas utilizadas. Una solución alternativa al problema de la masividad sería la implementación de prácticas de laboratorio en línea, utilizando un simulador de redes. Esto permitiría brindar al estudiante la posibilidad de tener más instancias prácticas y de estudiar topologías más complejas.

En esta línea los docentes Víctor González Barbone y Pablo Belzarena del Instituto de Ingeniería Eléctrica realizaron el trabajo “Incorporación De Un Simulador Gráfico De Redes En Un Objeto De Aprendizaje Reutilizable” presentado en el congreso del TAEE06 [6] [32]. En el mismo, los autores destacan las ventajas que puede tener la realización de prácticas de laboratorio en línea. En dicho trabajo, fue implementado y puesto a prueba un laboratorio con enfoque *e-learning*, que utiliza el simulador de redes EasySim1 [14].

EasySim es un simulador gráfico de redes, programado en Java, originado en un proyecto de fin de carrera de ingeniería eléctrica [14] y siendo retomado en otro proyecto posterior el cual desarrolló una segunda versión del simulador (EasySim2 [3]). Este simulador es fácil de usar, no obstante posee algunos errores de funcionamiento. Es así que, partiendo de los trabajos anteriores, este proyecto se plantea los objetivos que se describen a continuación.

1.1 Objetivos

El objetivo central de este proyecto de grado es el desarrollo de un material de apoyo pedagógico enfocado a redes de datos. Esencialmente, se proponen dos productos claramente diferenciados, a saber, un simulador gráfico de redes y una serie de prácticas de laboratorio enfocados a *e-learning*.

En lo que refiere al simulador, se traza como objetivo tener un simulador de fácil uso y bajo costo. Para esto, se parte de EasySim2 [3], un simulador ya existente y que es propiedad de la Universidad de la República. El mismo se caracteriza por tener una interfaz gráfica muy amigable, pero tiene errores de funcionamiento. Se plantea entonces realizar una revisión sobre el mismo y corregir los errores que en él se encuentren.

Las prácticas de laboratorio son el objetivo final del proyecto, ya que en estos se usará el simulador revisado. Se trata de una serie de prácticas de laboratorio con enfoque a *e-learning*, que pueden ser realizados en línea, para cursos relacionados

con las redes de datos. Los mismos se realizan bajo el estándar SCORM (*Shareable Content Object Reference Model*) [31] para empaquetamiento de objetos de aprendizaje.

1.2 Resultados

Se espera obtener los siguientes resultados:

- Simulador EasySim3, nueva versión del simulador gráfico de redes sin errores de funcionamiento, pero manteniendo la interfaz gráfica y arquitectura de EasySim2.
- Manual rápido de usuario para EasySim3
- Documento de definición de formatos para el diseño de prácticas de laboratorio.
- Cuatro prácticas de laboratorio en formato SCORM tocando distintos temas del área de redes de datos previamente acordados con los docentes de los cursos correspondientes

1.3 Metodología de trabajo

El trabajo se realiza siguiendo una metodología de trabajo iterativa e incremental, que consta de tres fases principales. Durante la primera fase, se estudian los trabajos previos, la documentación de los programas a utilizar, y se repasan los conocimientos del área de redes de datos. La segunda fase corresponde a la corrección de errores en los algoritmos de EasySim2. Esta se divide en tres etapas: la detección de errores, la corrección, y el testeo. Aunque en realidad estas etapas no se ordenan cronológicamente, ya que conforman un proceso iterativo. En esta etapa además, se realiza el manual rápido para el simulador EasySim3. Durante la tercera fase se desarrollan las prácticas de laboratorio, basados en los temas acordados, y respetando el formato previamente definido.

Se mantuvieron reuniones con los tutores periódicamente las cuales buscaban, en principio, definir claramente el alcance del proyecto, y luego mostrar el avance del proyecto y discutir las decisiones importantes. Además, se realizaron dos presentaciones de avance del trabajo dentro del marco del curso de gestión de proyecto.

Una vez generado el primer laboratorio completo en formato SCORM, el mismo fue presentado ante los docentes del área de redes del Instituto de Ingeniería Eléctrica. En esa instancia, se buscó obtener una evaluación del trabajo realizado y se definieron los temas de las restantes prácticas de laboratorio. Los resultados de esta instancia de discusión se presentan en el Capítulo 6.

El equipo de trabajo mantuvo reuniones semanales para la discusión de temas y la toma de decisiones, así como para la coordinación de las actividades y la evaluación de los resultados. Más del sesenta por ciento de las tareas fueron realizadas en conjunto, lo que permitió una discusión constante de cada punto. Cabe notar que, en la etapa de corrección de errores del *software* se aplicaron los conceptos de programación extrema [13] [28].

1.4 Organización del informe

El presente informe está organizado en siete capítulos que describen los aspectos más relevantes del trabajo realizado durante el desarrollo de este proyecto de grado. Se anexa además el material complementario para que el lector pueda profundizar en los temas tratados. Los capítulos a seguir están organizados de la siguiente forma:

Capítulo 1: Introducción. El presente capítulo busca interiorizar al lector en los lineamientos generales del proyecto. Se provee una breve descripción de los objetivos, resultados y metodología de trabajo, así como la organización del informe.

Capítulo 2: Marco de Trabajo. En este capítulo se brinda la respuesta a una pregunta básica ¿cómo se hacen las cosas hoy en el área en que estamos trabajando? En particular, ¿cómo se implementan actualmente los laboratorios de redes de datos? Además, se brinda una breve descripción del funcionamiento de cada uno de los cursos de las asignaturas correspondientes a esta materia. Luego, se busca definir algunos de los conceptos de redes que se utilizarán a lo largo de este trabajo.

*Capítulo 3: Conceptos preliminares. S*e encuentran aquí las definiciones de los conceptos necesarios para la comprensión del trabajo. En particular, en el área de educación semi-presencial. Entre los que se encuentran *e-learning*, SCORM, y Moodle.

Capítulo 4: Gestión del Proyecto. Este capítulo está dedicado a describir el manejo del proyecto. Se comienza por describir la metodología usada para la realización de la planificación. Se brinda un detalle de las tareas planificadas y los tiempos estimados. Se presenta el desarrollo real del proyecto, justificando desviaciones y explicando las medidas tomadas. Finalmente, se presenta una comparación entre la planificación y el desarrollo.

Capítulo 5: Análisis y Diseño de la Solución. Este capítulo se dedica a la etapa de estudio del proyecto. En él se puede encontrar una descripción del simulador EasySim2, detalles sobre la implementación de los protocolos TCP y OSPF. Además, se examinan requerimientos y diseño de las prácticas de laboratorio virtuales.

Capítulo 6: Desarrollo del proyecto. Este capítulo divide su contenido en dos secciones. Por un lado se presentan las implementaciones de las soluciones propuestas para la corrección del simulador. Mientras que por el otro, se describe el desarrollo de las etapas vinculadas con las prácticas de laboratorio.

Capítulo 7: Resultados y Conclusiones. En este capítulo se brinda una descripción de los resultados obtenidos a lo largo del proyecto. Luego se expresan las conclusiones finales del trabajo. Éstas se dividen en dos clases, las conclusiones técnicas y las conclusiones generales.

Capítulo 2: Marco de Trabajo

Este proyecto surge en medio de un entorno de cambio. El Instituto de Ingeniería Eléctrica (IIE de aquí en adelante) desde hace dos años está migrando las páginas web de sus cursos de un sitio web a un Sistema de Gestión de Contenidos Educativos (Moodle). Este sistema permite que las páginas no sean simplemente un lugar donde colocar información accesible a los estudiantes, sino que brinda una serie de herramientas para el manejo integral de un curso, tareas, cuestionarios, sistema de bedelía, etc. (Ver LCMS en el Capítulo 3)

Por otro lado se tiene la situación actual de la educación, cursos superpoblados, recursos físicos limitados en tiempo y espacio, y recursos económicos insuficientes. Es así que se genera la inquietud por unir al proceso de cambio del IIE un proyecto de reestructura para el área de redes de datos. En la siguiente sección, se brinda una breve descripción del funcionamiento de los cursos de las asignaturas correspondientes a esta materia actualmente. Luego, se busca definir los conceptos de redes que se utilizarán a lo largo de este trabajo.

2.1 Descripción del área de trabajo

Los cursos de área de redes que se dictan en el IIE son Redes de Datos, Evaluación de Performance en Redes de Telecomunicaciones, Redes Corporativas, y Ruteo IP y Tecnologías de Transporte. A continuación se brinda un resumen de cada asignatura y su metodología de trabajo.

2.1.1 Redes de datos

Redes de datos [24] es una asignatura de introducción al área de redes dictada como curso de grado para todos los perfiles de Ingeniería Eléctrica. La misma se plantea los siguientes objetivos:

- comprender y manejar los conceptos fundamentales de las redes de datos.
- fundamentar la necesidad del modelo de capas.
- definir para cada capa objetivos, funciones e interrelación entre capas.
- describir los principales protocolos de cada capa, sus características y ámbito de aplicación.
- describir y analizar ejemplos de redes usados en la realidad.

Metodología:

El curso se articula en clases teóricas y prácticas de laboratorio. Se jerarquizará especialmente la comprensión conceptual de los temas y su aplicación a situaciones de la realidad.

“Se intentará desarrollar en el alumno la comprensión crítica de los temas, habilitándolo para juzgar, ante casos concretos, las posibilidades de aplicación de diferentes soluciones técnicas, evaluando comparativamente ventajas, dificultades de implementación, costos y demás aspectos propios de la ingeniería de comunicaciones. El logro de estos objetivos requiere el conocimiento cabal de las diversas soluciones técnicas existentes en la actualidad, lo cual formará parte del contenido informativo del curso.”

Ganancia de curso y Evaluación

Se realizan dos parciales en el semestre. Cada parcial tiene un máximo de 50 puntos. Para exonerar o aprobar el curso en necesario aprobar el laboratorio, el cual consta de ocho prácticas.

Para aprobar el laboratorio es necesario:

- la asistencia al 80% de las clases de laboratorio
- la aprobación de los pre-informes individuales
- la aprobación de los informes grupales

2.1.2 Evaluación de Performance en Redes de Telecomunicaciones

Asignatura que puede ser cursada como curso de grado, pero es en realidad un curso de postgrado y actualización. Según se indica en la página del curso [12], el objetivo de mismo es: “El curso habilitará a los estudiantes a entender los principales factores que tienen influencia sobre la performance de las redes conmutadas de paquetes, que soportan datos y aplicaciones multimedia. Permitirá predecir el impacto de estos factores mediante métodos analíticos o simulados y a comparar estos resultados con la medición real de performance en una red. En el curso se realizará especial énfasis en herramientas para el estudio de performance en nuevas tecnologías de Internet como MPLS y diffserv.”

Metodología

El curso se articula en clases teóricas, prácticas de laboratorio y clases de consulta para el trabajo final. Se recorren los temas del curso, brindando las bases para que el estudiante pueda decidir su área de interés para la realización de la investigación final. Durante el desarrollo del curso se realizan prácticas de laboratorio obligatorias, en las que el estudiante debe entregar informes grupales con las observaciones realizadas. El trabajo final tiene una duración aproximada de un mes, durante el cual no se dictan clases teóricas, pero si hay clases de consulta.

Ganancia del curso y Evaluación

El curso se aprueba mediante la realización de laboratorios, entregas de ejercicios y la realización de un trabajo final individual en el que el estudiante desarrollará, con mayor profundidad, alguno de los temas que se ven en el curso. Este trabajo final debe presentarse ante un tribunal evaluador.

2.1.3 Redes Corporativas

Este es un curso de postgrado y actualización, el mismo tiene como objetivo presentar una visión de la tecnología de telecomunicaciones disponible en el mercado empresarial. Se abarcan las áreas de redes de voz, redes de datos, unificación de redes y soporte físico de las redes (cableado estructurado). Se trata de ver cómo se utilizan o pueden utilizarse dichas tecnologías en las empresas [25].

Metodología de Enseñanza:

Clases teóricas

2.1.4 Ruteo IP y tecnologías de transporte

Al igual que el curso de Performance de redes, Ruteo IP [30] puede ser cursado como curso de grado, pero es en realidad un curso de postgrado y actualización. Su objetivo es profundizar los conocimientos de redes de datos particularmente en protocolos de ruteo dinámico y tecnologías utilizadas como transporte en capa 2 (*Frame Relay*, ATM, MPLS).

Metodología de Enseñanza

Clases teórico-prácticas de aproximadamente 2 horas. Dos instancias de laboratorios. Los mismos son de nivel demostrativo, para ayudar a comprender conceptos impartidos durante el curso.

Ganancia del curso y Evaluación

El curso se aprueba mediante dos parciales. Para aprobar se requiere el 60% del puntaje total.

2.1.5 Estado actual

De las asignaturas antes descritas nos centraremos en Redes de Datos y Evaluación de Performance en Redes, por ser las que cuentan con prácticas de laboratorio obligatorias. Estas prácticas se llevan a cabo en el Laboratorio de desarrollo de *software* del IIE. Dicho laboratorio cuenta con doce computadoras en una red local, lo que permite generar topologías de hasta doce enrutadores si cada computadora emula un enrutador. Esta es una limitante a la hora de diseñar los laboratorios. Además, los grupos generalmente superan las treinta personas, debiendo los estudiantes compartir los puestos de trabajo de a dos o hasta tres por computadora.

Los laboratorios se dividen en tres etapas. La primera es de preparación, en la que el estudiante debe leer el material que se indique y completar un cuestionario previo. La segunda es la instancia de práctica. En esta los estudiantes concurren al laboratorio, en día y hora previamente fijados, y mediante un procedimiento realizan la práctica y toman las observaciones correspondientes. La tercera etapa puede darse en la misma instancia que la segunda o durante los días posteriores. En ella debe llenarse un formulario de evaluación con los resultados obtenidos el que se entrega al docente para su corrección.

Tanto el cuestionario previo, como el posterior son tomados por los docentes como evaluación. El primero es revisado durante la práctica y el segundo una vez terminada la misma y entregado. La realización de los cuestionarios en forma individual o grupal puede variar, aunque en el caso del cuestionario posterior es más difícil evaluar en forma individual ya que se realiza una tarea grupal.

En nuestra opinión, en todo curso de redes y en particular en los cursos introductorios es fundamental contar con prácticas. Las mismas permiten que los alumnos prueben y vean en funcionamiento mecanismos que en general son invisibles. El trabajo de un protocolo o de los algoritmos de enrutamiento, son conceptos teóricos y abstractos, difíciles de plasmar en la realidad. Por lo tanto, consideramos que la realización de más prácticas contribuiría a aclarar dichos conceptos.

La dinámica que se posee para la realización de práctica, limita el tiempo que el alumno puede dedicar a experimentar. El laboratorio es pequeño y de uso compartido entre distintas asignaturas. Incluso durante el tiempo dedicado a la práctica, los puestos de trabajo deben ser compartidos. La duración del semestre y la cantidad de temas a abordar, hacen que las prácticas no sean tantas como los docentes desean, ni lo suficientemente profundas. En las tres horas que dura una práctica promedio, deben observarse varios mecanismos diferentes, por lo que el tiempo para experimentar se limita más aún.

2.2 TCP

TCP (*Transmisión Control Protocol*) [5] es un protocolo orientado a conexión y confiable a nivel de capa de transporte. Fue diseñado específicamente para proporcionar una corriente de Bytes en forma confiable a través de Internet. La misma se diferencia de una red simple ya que sus distintas partes pueden tener altos grados de diferencia en lo que respecta a topologías, anchos de banda, retardos, tamaños de paquetes y otros parámetros. TCP fue concebido con el propósito de adaptarse dinámicamente a los cambios en Internet, presentándose robusto ante diversos tipos de fallas de la misma.

TCP fue definido formalmente en la RFC 793. Con el pasar del tiempo se detectaron errores varios e inconsistencias, lo cual trajo aparejado cambios en algunas áreas. Dichas fallas y sus correspondientes correcciones se detallan en la RFC 1122. En la RFC 1323 se presentan algunas extensiones realizadas.

Un equipo que maneja TCP, posee una entidad de transporte TCP y una interfaz con la capa de red (capa IP). Dicha entidad acepta corrientes de datos del usuario, los divide en partes que no excedan los 64 KBytes (generalmente 1500 Bytes), y envía cada parte como un datagrama IP independiente. Al arribar a otra máquina, los datagramas IP que contienen datos TCP son entregados a la entidad TCP, la cual reconstruye las corrientes originales de Bytes.

Dado que la capa de red no ofrece garantía alguna de entrega de los datagramas, es responsabilidad de la entidad TCP temporizar los envíos y retransmitir los segmentos en caso de ser necesario. Otra responsabilidad de la entidad TCP es reensamblar los mensajes con la secuencia adecuada, dado que los datagramas IP pueden llegar desordenados. En pocas palabras, TCP debe proveer la confiabilidad que los usuarios pretenden y que el protocolo IP no proporciona.

2.3 OSPF

OSPF (*Open Shortest Path First*) [5] [33] [34] es un protocolo de ruteo interno. Fue definido originalmente en la RFC 1131 y perfeccionado en la RFC 2328, Internet Standard 54. Su finalidad es armar las tablas de ruteo que se usan para encaminar paquetes a través de redes pequeñas, conocidas como Sistemas Autónomos (AS de *Autonomous Systems*). Actualmente, OSPF es el protocolo de enrutamiento interior más usado, principalmente cuando se interconectan equipos de diferentes fabricantes.

OSPF calcula los caminos de menor costo entre todos los enrutadores que componen el AS. La información con la cual cuenta cada enrutador para llevar a cabo el protocolo es la que le envían los demás enrutadores del AS, sobre cual es el costo

para acceder a sus vecinos inmediatos[1]. Luego, con la información obtenida se confecciona una tabla que contiene información de toda la red. OSPF se vale del algoritmo de Dijkstra para efectuar el cálculo de los "mejores caminos". Dicho algoritmo se corre sobre la tabla recientemente mencionada. Como resultado de los pasos anteriores, el enrutador obtiene su tabla de ruteo. Cabe destacar que este proceso es dinámico, por lo tanto responde de manera eficiente a cambios en la topología de la red, recalculando las rutas de ser necesario.

2.4 Simulador de redes de datos

Un simulador de redes es un *software* que permite simular o emular el comportamiento de los elementos de una red de datos, como ser terminales (*host*), enrutadores, conmutadores, concentradores, etc. Entre las cualidades deseables de un simulador para enseñanza de redes se destacan:

- facilidad de manejo, tanto para el armado de topologías como para la configuración de parámetros.
- visualización gráfica:
    - de la topología.
    - de la variación de valores en distintos puntos (pérdidas, largo de cola, tiempo de retorno).
    - de rutas, trayectorias de paquetes, pérdidas.
- multiplataforma, no debe depender del sistema operativo del alumno.
- fácil de instalar.
- nulo o muy bajo costo.

Algunos simuladores conocidos son:

CNet v2.0.10: orientado a enseñanza, pero no corre en MS Windows. Además requiere que el alumno compile y edite código para su instalación.

Ns2: orientado a investigación en redes de datos. Deben bajarse diversos paquetes y las fuentes para compilar. Requiere conocimientos de C++ y OTcl. Las animaciones requieren el paquete adicional nam, Network Animator. Ns es universalmente aceptado como simulador de prueba para investigación en redes, pero resulta poco apto para la enseñanza.

NCTUns: es un simulador y emulador de redes. Corre solo sobre sistemas Linux ya que hace uso de su stack de protocolos, por lo que requiere la compilación e instalación de un nuevo kernel.

OpNet: originado en un desarrollo en el MIT (USA), de uso comercial. Ofrece múltiples capacidades de simulación, animación y análisis. Sin restricción de plataforma.

CCNA: para auto-estudio, de Cisco, es un producto comercial orientado al manejo de los equipos de sus fabricantes y a los exámenes de certificación.

Ninguno de los ejemplos anteriores cumple las condiciones anotadas antes respecto a simuladores para aprendizaje, ya sea por sus dificultades de instalación

[1] Entiéndase por "vecino inmediato" a aquellos enrutadores conectados a un enrutador por medio de un enlace directo.

masiva por los estudiantes, por razones técnicas o por restricciones de licencia y los costos involucrados [6].

2.4.1 EasySim3

EasySim es un simulador programado en lenguaje Java, lo cual permite que sea utilizado sobre cualquier tipo de plataforma o de Sistema Operativo. El único requerimiento es que se cuente con una máquina virtual java (*Java Virtual Machine*, JVM) compatible instalada. El simulador puede bajarse fácilmente por la red en forma de *applet* o de programa ejecutable. Además, el simulador de eventos discretos en que se basa es el *software* de dominio público SimJava

La interfaz de configuración resulta muy amigable para construir la red: agregar *hosts*, enlaces, enrutadores, y configurar cada uno de estos componentes es muy sencillo. Los resultados de la simulación se presentan gráficamente y a través de informes, lo cual facilita el análisis.

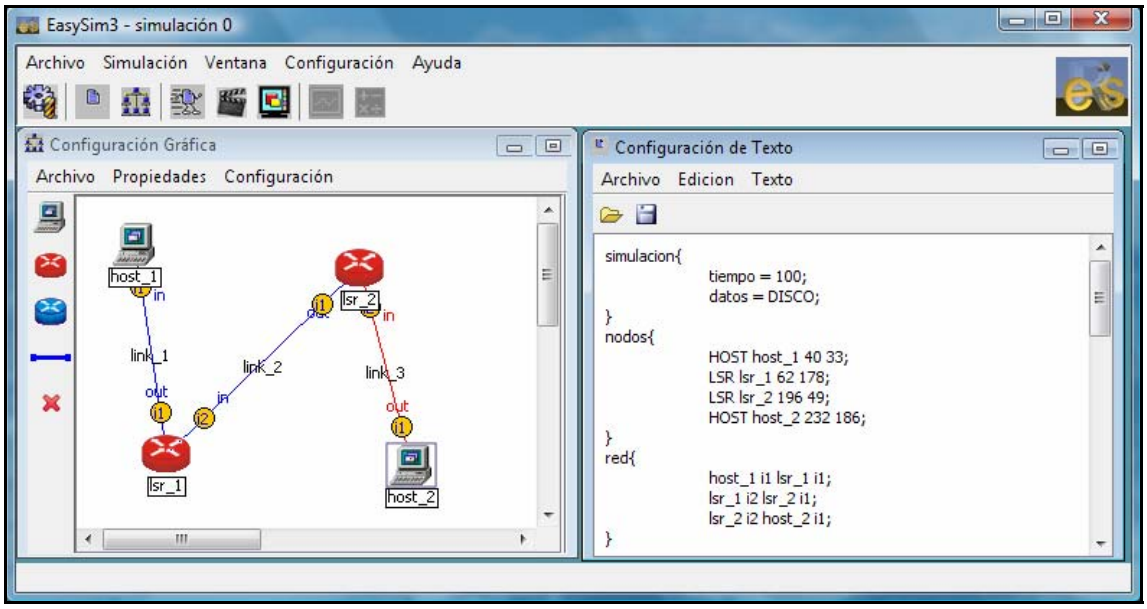


Figura 1: Interfaces de configuración de EasySim3, Gráfica (izquierda) y Texto (derecha)

Por la arquitectura del simulador se pueden utilizar diferentes mecanismos de establecimiento de rutas, protocolos de reserva de recursos, y procesar los paquetes que arriban a los nodos. Además, se pueden definir la interfaz de salida en base a la ruta y clase de servicio, enviarlo a la cola correspondiente, aplicarle diferentes políticas de descarte y despacho de paquetes por la interfaz.

Los diferentes tipos de generadores de paquetes disponibles en un *host* entregan los paquetes a diversos agentes. La implementación actual incluye: agentes UDP, TCP Reno, TCP Vegas y Fast TCP. Los agentes envían los paquetes a la interfaz correspondiente, la cual a su vez los enviará a los nodos intermedios hasta llegar al *host* en el otro extremo de la red.

EasySim permite simulaciones animadas (ver Figura 2) en las que es posible visualizar el movimiento de los paquetes, el estado de las colas, el estado de los enlaces, las tablas de ruteo, y las tablas de recursos entre otras variables. Para redes muy grandes, en que las simulaciones pueden tornarse pesadas, puede ser más conveniente realizar una simulación simple y visualizar los resultados en gráficas que proporciona el propio simulador.

Finalmente, un detalle no menor es que este simulador de redes es propiedad de la Universidad de la República y por lo tanto es gratis.

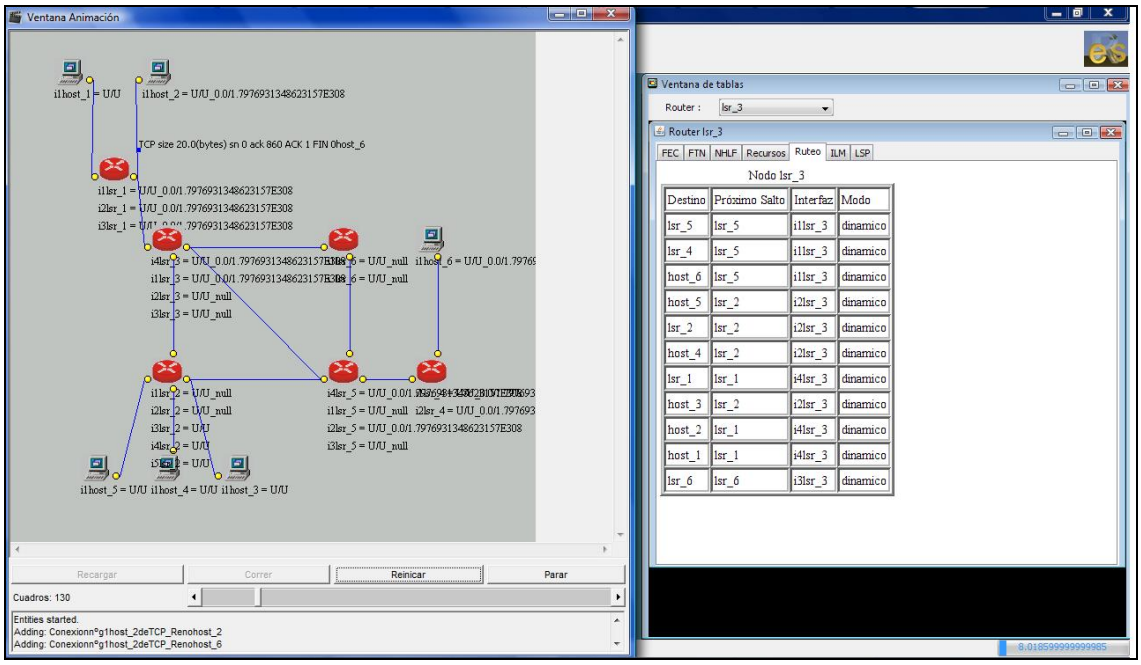


Figura 2: Simulación animada, visualizando una tabla de ruteo

Capítulo 3: Conceptos preliminares

Los objetivos del proyecto y la línea de trabajo seguida, determinan que las prácticas de laboratorio se diseñen con enfoqué *e-learning,* pero ¿qué es *e-learning*? En este capítulo se brinda información básica sobre este concepto y otros relacionados al tema de la educación semipresencial.

3.1 E-learning

El concepto de *e-learning* se define de muchas formas diferentes, fundamentalmente debido a que los actores que de él hacen uso son muy diversos, cada uno con su idiosincrasia y su ámbito de aplicación [15].

Desde la perspectiva de su concepción y desarrollo como herramienta formativa, los sistemas de *e-learning* tienen una dualidad pedagógica y tecnológica. Pedagógica en cuanto a que estos sistemas no deben ser meros contenedores de información digital, sino que ésta debe ser transmitida de acuerdo a modelos y patrones pedagógicamente definidos para afrontar los retos de estos nuevos contextos. Tecnológica en cuanto a que todo el proceso de enseñanza-aprendizaje se sustenta en *software*, principalmente desarrolladas en ambientes web, lo que le vale a estos sistemas el sobrenombre de plataformas de formación.

Desde la perspectiva de su uso se podría distinguir la visión que tienen sus usuarios finales, que con independencia de su madurez y formación, verán al sistema *e-learning* como una fuente de servicios para alcanzar su cometido formativo. No obstante, también es factible diferenciar una visión de organización. En esta, se definen el alcance y los objetivos buscados con la formación basada en estos sistemas, distinguiéndose una visión académica y una visión empresarial.

Si se toma como referencia la raíz de la palabra, *e-learning* se traduce como “aprendizaje electrónico”, y como tal, en su concepto más amplio puede comprender cualquier actividad educativa que utilice medios electrónicos para realizar todo o parte del proceso formativo.

Existen definiciones que abren el espectro del *e-learning* a prácticamente a cualquier proceso relacionado con educación y tecnologías, como por ejemplo la definición de la *American Society of Training and Development* que lo define como:

> *“término que cubre un amplio grupo de aplicaciones y procesos, tales como aprendizaje basado en web, aprendizaje basado en ordenadores, aulas virtuales y colaboración digital. Incluye entrega de contenidos vía Internet, intranet/extranet, audio y vídeo grabaciones, transmisiones satelitales, TV interactiva, CD-ROM y más”.*

Otros autores acotan más el alcance del *e-learning* reduciéndolo exclusivamente al ámbito de Internet, como Rosenberg (2001) que lo define como:

> *“el uso de tecnologías Internet para la entrega de un amplio rango de soluciones que mejoran el conocimiento y el rendimiento. Está basado en tres criterios fundamentales: 1. El e-learning trabaja en red, lo que lo hace capaz de ser instantáneamente actualizado, almacenado, recuperado, distribuido y permite compartir instrucción o información. 2. Es entregado al usuario final a través del uso de ordenadores utilizando tecnología estándar de Internet. 3. Se enfoca en la visión más amplia del aprendizaje que van más allá de los paradigmas tradicionales de capacitación”.*

Según García Peñalvo [15], se define *e-learning* como:

"la capacitación no presencial que, a través de plataformas tecnológicas, posibilita y flexibiliza el acceso y el tiempo en el proceso de enseñanza-aprendizaje, adecuándolos a las habilidades, necesidades y disponibilidades de cada discente, además de garantizar ambientes de aprendizaje colaborativos mediante el uso de herramientas de comunicación síncrona y asíncrona, potenciando en suma el proceso de gestión basado en competencias."

En todas estas definiciones, así como en otras que se pueden encontrar, se acaba haciendo mención explícita o implícita a lo que se viene llamando en triángulo del *e-learning* (Lozano, 2004), formado por los siguientes elementos:

- Tecnología: plataformas de aprendizaje.
- Contenidos: la calidad y estructuración de los contenidos.
- Servicios o herramientas: siendo el elemento más heterogéneo que engloba la acción de los profesores, elementos de gestión, elementos de comunicación, elementos de evaluación.

Variando el peso de estos tres componentes se obtienen diferentes modelos de formación electrónica, de igual forma que variando las variables y recursos con los que cuenta un profesor se obtienen diferente políticas de docencia presencial. Es frecuente encontrar cursos en línea en los que sus contenidos no pasan de ser mera virtualización de cursos previos en los que el alumno lee ahora en pantalla lo que antes podía leer en papel. Esto es más *e-reading* que *e-learning*. El diseño de los contenidos debe de ser realizado con el objetivo de que respondan a:

- adecuación a las necesidades y posibilidades del alumno
- calidad y cantidad de la información presentada
- interactividad
- estructura adecuada para su correcta asimilación

Buscando cumplir con éstos objetivos se definen los objetos de aprendizaje como forma de estructurar el contenido, y en asociación a ellos los metadatos. Se le llama objeto de aprendizaje a cualquier recurso, digital o no, basado en un solo objetivo de aprendizaje que puede ser utilizado, reutilizado o referenciado en el proceso de enseñanza. Estos objetos pueden ser editados, adaptados o recombinados de acuerdo con propósitos específicos. Para cumplir con esta definición se pide que los objetos de aprendizaje cumplan con ciertas condiciones: reusabilidad, accesibilidad, interoperabilidad y durabilidad [29].

Si se trata de armar los contenidos con objetos de aprendizaje como si se armara un muro con bloques, es necesario tener bien identificado cada bloque. Esta es precisamente la principal función de los metadatos. Los metadatos son un conjunto de datos que se incluyen a un objeto de aprendizaje para identificarlo. Existe el estándar LOM (*Learning Object Metadata*) de la IEEE [20], que aunque no es el único, es el que define la mayor cantidad de metadatos. Entre las cualidades relevantes de los objetos de aprendizaje que se describen incluyen: título, idioma, tipo de objeto, autor, propietario, términos de distribución, formato, *copyright*, y cualidades pedagógicas, tales como estilo de la enseñanza o de la interacción. Con esto se busca conocer todo lo necesario de un objeto de aprendizaje sin tener que poseer el objeto mismo.

En la práctica, para llevar a cabo un programa de formación basado en *e-learning*, se hace uso de plataformas o sistemas de *software* que permiten la comunicación e interacción entre profesores, alumnos y contenidos. Se tienen principalmente dos tipos de plataformas: las que se utilizan para impartir y dar seguimiento administrativo a los cursos en línea o LMS (*Learning Management Systems*) y, por otro lado, las que se utilizan para la gestión de los contenidos digitales o LCMS (*Learning Content Management Systems*).

3.2 LCMS

Entre las herramientas más utilizadas para los ambientes o sistemas *e-learning* están los Sistemas de Administración de Aprendizaje o LMS, también ampliamente conocidos como plataformas de aprendizaje. Un LMS es un *software* basado en un servidor web que provee módulos para los procesos administrativos y de seguimiento que se requieren para un sistema de enseñanza, simplificando el control de estas tareas. Los módulos administrativos permiten, por ejemplo, configurar cursos, matricular alumnos, registrar profesores, asignar cursos a un alumno, llevar informes de progreso y calificaciones. También facilitan el aprendizaje distribuido y colaborativo a partir de actividades y contenidos preelaborados, utilizando los servicios de comunicación de Internet como el correo, los foros, las videoconferencias o el *chat*.

El alumno interactúa con la plataforma a través de una interfaz web que le permite seguir las lecciones del curso, realizar las actividades programadas, comunicarse con el profesor y con otros alumnos, así como dar seguimiento a su propio progreso con datos estadísticos y calificaciones. La complejidad y las capacidades de las plataformas varían de un sistema a otro, pero en general todas cuentan con funciones básicas como las que se han mencionado. Entre las plataformas comerciales más comunes se encuentran Blackboard [7] y WebCT [35], mientras que las más reconocidas por parte del *software* libre son Moodle [22] y Claroline [9].

Los Sistemas de Administración de Contenidos de Aprendizaje o LCMS tienen su origen en los CMS (*Content Management System*) cuyo objetivo es simplificar la creación y la administración de los contenidos en línea, y han sido utilizados principalmente en publicaciones periódicas (artículos, informes, fotografías…). En la mayoría de los casos lo que hacen los CMS es separar los contenidos de su presentación y también facilitar un mecanismo de trabajo para la gestión de una publicación web. Los LCMS siguen el concepto básico de los CMS, que es la administración de contenidos, pero enfocados al ámbito educativo, administrando y concentrando únicamente recursos educativos y no todo tipo de información.

El proceso de trabajo dentro de un LCMS requiere de control en cada fase del contenido, esto conlleva un proceso editorial para controlar la calidad de los contenidos creados, así como para permitir y organizar su publicación.

3.2.1 Moodle

Moodle en un sistema de gestión de la enseñanza (CMS o LCMS), es decir una aplicación diseñada para ayudar a los educadores a crear cursos de calidad en línea. Moodle fue creado por Martin Dougiamas, quien trabajó como administrador de WebCT en la Universidad Curtin, y se basó en trabajos sobre el constructivismo en pedagogía, que afirman que el conocimiento se construye en la mente del estudiante en lugar de ser transmitido sin cambios a partir de libros o enseñanzas. Un profesor

que opera desde este punto de vista, crea un ambiente centrado en el estudiante que lo ayuda a construir ese conocimiento en base a sus habilidades y conocimientos propios en lugar de simplemente publicar y transmitir la información que consideran que los estudiantes deben conocer [22].

Moodle ha venido evolucionando desde 1999 y nuevas versiones siguen siendo producidas. En enero de 2005, la base de usuarios registrados incluye 2600 sitios en más de 100 países y está traducido a más de 50 idiomas. El sitio más grande reporta tener actualmente 6.000 cursos y 30.000 estudiantes [23].

Este LCMS se distribuye gratuitamente como *software* libre (*Open Source*) (bajo la Licencia Pública GNU). Esta licencia pretende garantizar la libertad de compartir y modificar *software* libre, para asegurar que el *software* es libre para todos sus usuarios. Asimismo, da permiso legal para copiar, distribuir y/o modificar el *software*, siempre que se respeten tanto los derechos como las obligaciones inherentes. Cuando se habla de *software* libre, se refiere a libertad, no a precio. Las licencias GNU están diseñadas para asegurar de que se tenga la libertad de distribuir copias de *software* libre (y cobrar por ese servicio si se quiere). Además, de que reciba el código fuente o que pueda conseguirlo si lo quiere, de que pueda modificar el *software* o usar fragmentos de él en nuevos programas libres. Asimismo, asegura que se sepa que se pueden hacer todas estas cosas. Básicamente esto significa que Moodle tiene derechos de autor (*copyright*), pero se cuenta con la libertad de modificarlo, siempre que se respete su licencia.

Moodle era al principio un acrónimo de *Modular Object-Oriented Dynamic Learning Environment* (Entorno de Aprendizaje Dinámico Orientado a Objetos y Modular), lo que resulta fundamentalmente útil para programadores y teóricos de la educación. Además, Moodle puede funcionar en cualquier computadora en la que pueda correr PHP, y soporta varios tipos de bases de datos (en particular MySQL).

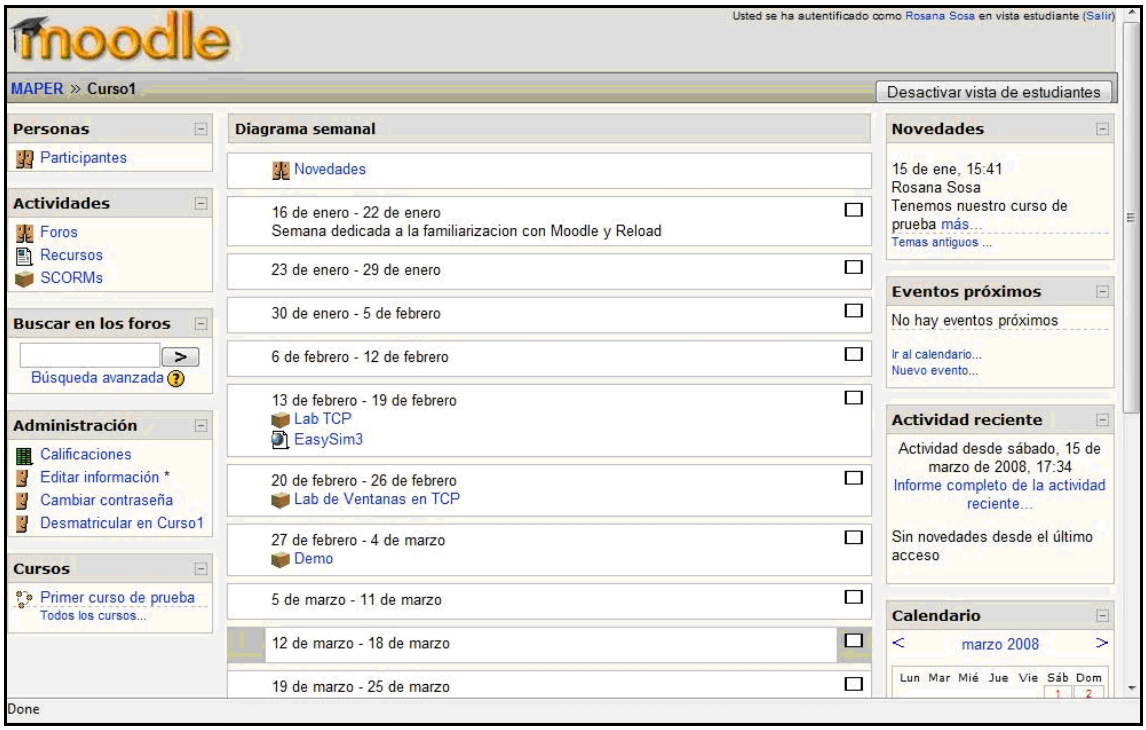


Figura 3: Curso en Moodle

3.3 SCORM

SCORM (*Shareable Content Object Reference Model*) es un estándar que permite al usuario creador de un curso, generar material para múltiples LCMS. Los Objetos de Aprendizaje en formato SCORM pueden ser portados entre distintas plataformas que cumplan con el estándar.

El empaquetamiento SCORM es un estándar de intercambio de materiales de Objetos de Aprendizaje generado por ADL (*Advanced Distribute Learning*) [1] para permitir la interoperabilidad, accesibilidad y reutilización de contenidos educativos en un contexto de Educación a Distancia. Permite además que el material electrónico pueda ser entendido por distintas plataformas (LCMS) e incluso transformado en un medio electrónico distinto y específico de la herramienta LCMS. Sin embargo para que un LCMS pueda manejar paquetes SCORM debe al menos implementar el *Run-Time Environment* especificado por SCORM.

Además toma en cuenta el grado de complejidad de los objetos de aprendizaje. Se pueden definir tanto elementos básicos (*Asset*) como puede ser un texto, una imagen, una página web o elementos que son conjuntos de elementos básicos (*SCO – Sharable Content Object*).

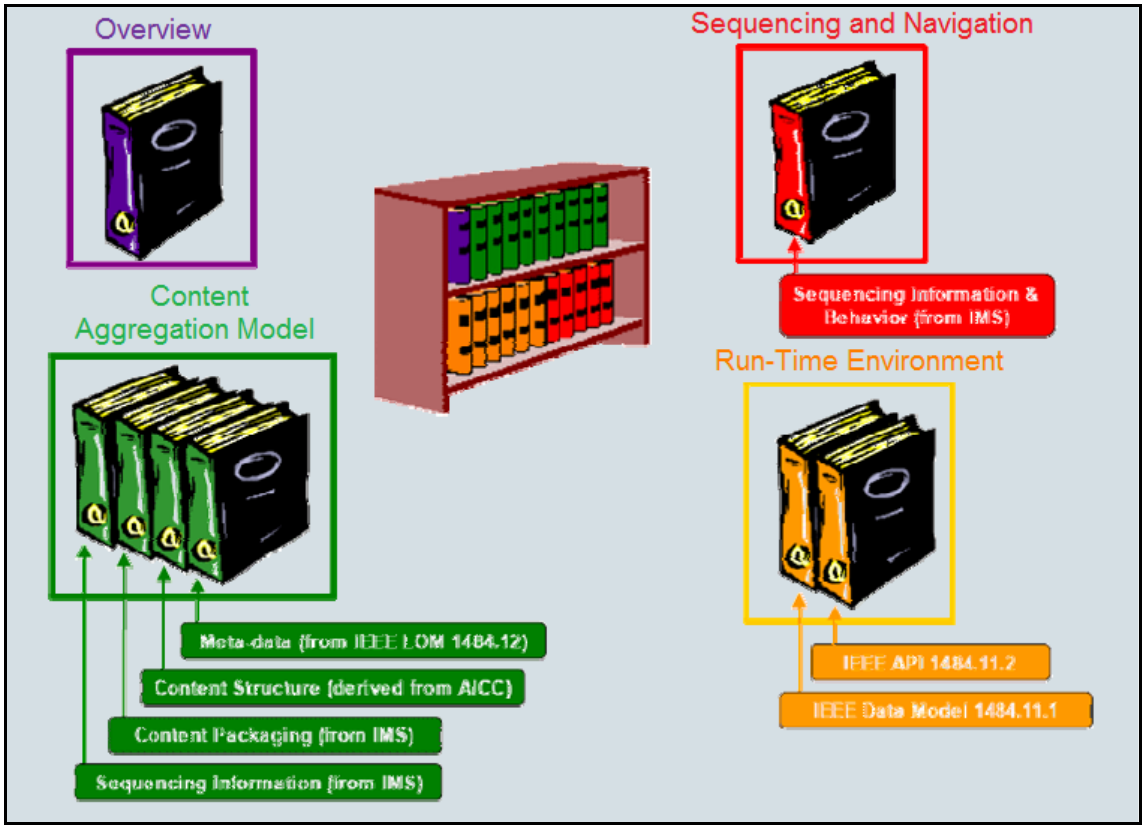


Figura 4: SCORM 2004 (Bookshelf)

La Figura 4 muestra un esquema de los distintos componentes del estándar SCORM.

Capítulo 4: Gestión del proyecto

Este apartado busca relatar como se realizaron las estimaciones de esfuerzo para el proyecto. Se describen los puntos tenidos en cuenta y se brinda un detalle de las horas planificadas para cada una de las tareas propuestas. Además, se mencionan las fechas propuestas para cada uno de los puntos de control de avance. Como en la mayoría de los proyectos ocurrieron desviaciones en los planes. Se justifica también en este apartado las desviaciones ocurridas y como fueron manejadas.

4.1 Planificación

La planificación fue realizada a modo de cumplir con los objetivos finales en tiempo y forma. El tiempo de duración del proyecto es de un año con lo que todo debió ser pensado para ese tiempo. No obstante, se preestablecieron dos puntos de control de avance. Dividiendo así el proyecto en tres etapas de aproximadamente cuatro meses cada una. Para tener un efectivo control de avance se debieron planificar entregables para cada uno de estos puntos.

Como se describió en el apartado “Objetivos” se tienen dos objetivos bien diferenciados, el simulador y los laboratorios. Dichos objetivos fueron tratados por separado en la planificación. A fin de realizar una estimación más acertada, cada uno fue descompuesto en una serie de tareas. Cada una de las cuales fue elegida de manera tal que el cumplimiento de ésta asegura el cumplimiento de un objetivo. Por cumplimiento del objetivo debe entenderse verificar un determinado criterio de éxito.

La metodología seguida para realizar la planificación constó de varios pasos. Se comenzó por determinar tareas para cada objetivo. Luego se asignó a cada una de estas un esfuerzo medido en horas. Además, se debió asignar a cada tarea una fecha de finalización que asegurara el cumplimiento del objetivo en el tiempo preestablecido. Por otro lado, se realizó una estimación de cual sería el esfuerzo dedicado por cada uno de los integrantes en cada etapa del proyecto (mes a mes). Teniendo en cuenta el esfuerzo necesario para cada tarea, su fecha de finalización y la dedicación de los integrantes en cada período, se realizaron las asignaciones de tareas.

A continuación se muestra como se aplicó la metodología antes descrita a cada etapa del proyecto. Se describen las tareas planificadas y las medidas preventivas tomadas para paliar eventuales retrasos (por ejemplo sobre estimar tareas). Por más detalle a cerca de la planificación ver Anexo G

4.1.1 El simulador

El criterio de éxito establecido para el simulador fue que éste respondiera en forma adecuada ante una prueba predefinida: la simulación que se lleva a cabo en el primer laboratorio de Performance de Redes. Cabe destacar que el objetivo era tener el simulador funcionando correctamente, más allá de la simulación elegida como criterio de éxito. Para lograr esto se eligieron como tareas las siguientes:

- Detección de errores
- Corrección de los errores encontrados
- Revisión

Para la fase de detección de errores se estimó un esfuerzo de 105 horas. Se planificó que todos los integrantes trabajarían en esta fase. No necesariamente desarrollando la misma actividad, ya que la tarea fue descompuesta en sub tareas. Entre estas se destacan familiarización con el simulador, revisión de algoritmos y evaluación en detección. Imponiendo la restricción de dedicación horaria de cada integrante se llegó a que esta fase debería ser cumplida en un período de dos meses.

La corrección de errores fue estimada en 160 horas. También en esta fase se planificó que el grupo trabajaría junto, dada la escasa experiencia en el área. Algunas de las sub tareas planificadas para esta fase fueron corrección de código, evaluación de comportamiento y documentación de cambios. Dado que para esta etapa se planificó aumentar la dedicación horaria de cada integrante, la misma tuvo una duración planificada de un mes.

La revisión fue una fase pensada para ser desarrollada en paralelo con las dos anteriores. O sea, el objetivo de esta tarea fue detectar errores y corroborar correcciones. Se establecieron como sub tareas pruebas de testeo, revisión de correcciones y generación de reportes. La duración estimada de esta fase fue de dos meses u 80 horas.

Cronológicamente, se planificó comenzar por una familiarización con el simulador. Luego de lo que se comenzarían a realizar pruebas y revisión de código en paralelo. Una vez detectados los errores se procedería a corregirlos, volviendo a realizar pruebas con objeto de asegurar el correcto funcionamiento. Después de esto se propuso realizar un testeo general y generar las documentaciones pertinentes.

Buscando dejar un margen de tiempo para contemplar imprevistos, se estimó en dos semanas la preparación de la presentación del primer entregable. La estimación para esta etapa fue de 4 meses de trabajo, dedicando un esfuerzo total de 420 horas. Toda esta planificación apuntó a que se llegara al fin de la primera etapa con el simulador corregido.

	Nombre de tarea	Duración	Comienzo	Fin
1	Detección	105 horas	mié 16/05/07	vie 20/07/07
2	Corrección	160 horas	lun 23/07/07	vie 24/08/07
3	Testeo	420 horas	mié 16/05/07	mié 05/09/07
4	Presentación p entregable 1	20 horas	mié 05/09/07	vie 14/09/07
5	**Entregable 1 Sim corregido**	0 días	sáb 15/09/07	sáb 15/09/07

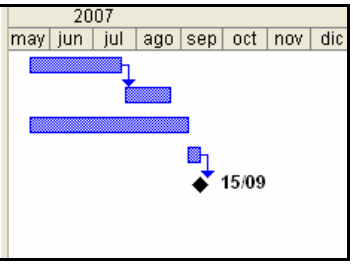


Figura 5: Diagrama de Gantt estimado para la primera etapa

4.1.2 Prácticas de laboratorio

En lo que tiene que ver con las prácticas de laboratorio el objetivo planteado fue implementar cuatro prácticas de laboratorio enfocadas a *e-learning*. La concretización de este objetivo fue planificada para la segunda etapa del proyecto, luego del primer punto de control de avance. Se asumió para la planificación de esta etapa que ya se contaría con el simulador corregido (en la primera etapa del proyecto). El criterio de éxito establecido a priori fue que se tuvieran los cuatro objetos de aprendizaje empaquetados bajo el estándar SCORM (y accesible desde la plataforma Moodle).

Análogamente a lo realizado en la etapa de corrección del simulador, esta etapa fue dividida en tareas. Dada la poca experiencia de los integrantes de este

grupo en el área didáctica, se pretendió obtener sugerencias por parte de docentes. A tal fin se propuso desarrollar un prototipo de laboratorio y obtener una evaluación del mismo. Se propusieron entonces las siguientes tareas para esta etapa:

- Elaboración de un manual rápido para el simulador
- Diseño e implementación del prototipo de laboratorio
- Definición de un formato para las prácticas de laboratorio
- Desarrollo de cuatro objetos de aprendizaje bajo el estándar SCORM

Teniendo en cuenta que el simulador sería usado para todas las prácticas y que el manual existente es relativamente extenso, se propuso desarrollar una guía rápida. Dicha guía debería apuntar a lectores con conocimientos básicos de redes. Brindando a estos una referencia corta y esencialmente gráfica de cómo llevar a cabo una simulación y observar resultados. Para esta tarea se estimo un esfuerzo de 15 horas. Al ser esta una tarea relativamente corta y auto contenida, fue asignada a un único integrante del proyecto.

Para implementar el prototipo de laboratorio se planteó elegir un tema y desarrollar una práctica. Se plantearon una serie de tareas a efectuar para el desarrollo de la práctica. Como ejemplo de estas se puede mencionar: selección de material teórico, elaboración de cuestionarios previos y posteriores, familiarización con el generador de paquetes SCORM, etc. Estas sub-tareas fueron divididas entre los integrantes del proyecto. Se estimó que el desarrollo del prototipo insumiría 70 horas. Imponiendo las restricciones de tiempo de dedicación de los integrantes, se consideró que el desarrollo del laboratorio tomaría tres semanas.

Una vez finalizada la implementación se pretendió obtener una evaluación. A tal fin se propuso que el prototipo de laboratorio fuera analizado por docentes y/o alumnos. El objetivo era obtener críticas y sugerencias en base a las cuales se corregiría el prototipo y se definirían formatos para las próximas prácticas de laboratorio. Para esto se establecieron algunas tareas como: coordinación de la evaluación con los docentes, procesamiento de los resultados obtenidos, implementación de mejoras, definición del formato, etc. Esta fase tuvo estimada una duración de 135 horas, distribuyéndose las tareas entre los integrantes del proyecto. Teniendo en cuenta el tiempo de dedicación de los integrantes, se estimó que esta fase quedaría cumplida en siete semanas.

Cumplida la fase anterior se debería tener un laboratorio implementado y un formato definido. Para cumplir con el objetivo propuesto se deberían desarrollar tres prácticas de laboratorio más. Como el proyecto cuenta con tres integrantes, se propuso que cada uno implementara una práctica. Luego de esto se haría una revisión interna de los laboratorios implementados (cada uno revisaría los hechos por los demás integrantes). Esta fase tuvo una duración estimada de 210 horas u 8 semanas imponiendo la restricción de tiempo de dedicación.

La planificación de estas fases fue hecha de manera tal que, a la fecha del segundo punto de control de avance ya se hubiesen cumplido todos los objetivos del proyecto. Esto fue hecho así buscando que se contara con un margen de tiempo que asegurara el cumplimento de los objetivos al final del proyecto.

	Nombre de tarea	Duración	Comienzo	Fin
1	Manual rápido	15 horas	lun 17/09/07	lun 24/09/07
2	Prototipo	70 horas	lun 17/09/07	mié 24/10/07
3	Definición de formato	135 horas	vie 05/10/07	mié 19/12/07
4	Laboratorios	210 horas	jue 20/12/07	vie 15/02/08
5	**Entregable 2**	0 horas	vie 15/02/08	vie 15/02/08

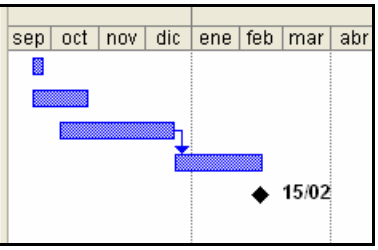


Figura 6: Diagrama de Gantt estimado para la segunda etapa

4.1.3 Documentación

Como se mencionó anteriormente, según la planificación hecha, al llegar a esta etapa ya se debería haber cumplido los criterios de éxito del proyecto. Con lo que esta etapa estaría dedicada a revisión del trabajo realizado y documentación. Las tareas que deberían realizarse en esta etapa serían las siguientes:

- Redacción en de un articulo en formato IEEE
- Diseño de un afiche para el proyecto
- Documentación general del proyecto

No fue realizada una planificación exhaustiva de esta etapa como fue hecho en las demás. Esto se debió a que ésta no contaba con tareas fácilmente predecibles como las demás. Por otro lado, se dejó un largo período de tiempo para esta parte a fin de paliar posibles retrasos.

4.2 Desviaciones y cronograma

Como se describió en el apartado anterior, todo el proyecto fue planificado de manera que se pudieran cumplir con los tiempos preestablecidos. No obstante, y como se podría esperar, no todo ocurrió de acuerdo a lo planificado. Algunas tareas tomaron menos tiempo de lo esperado, pero otras requirieron más tiempo de lo estimado a priori. A continuación, se presenta la evolución del proyecto a lo largo del tiempo y se justifican las desviaciones ocurridas. Se optó por detallar estas secciones en paralelo a fin de facilitar el entendimiento. Conjuntamente se describen las medidas tomadas para disminuir el efecto de dichas desviaciones en el cronograma inicial. Este apartado se divide en tres etapas de igual forma que el proyecto. Se mencionan las fechas en que ocurrieron los hechos más relevantes, buscando brindar una idea más gráfica de la evolución de los mismos.

4.2.1 Primera etapa

El proyecto comienza en Marzo de 2007. Lo primero que se debió realizar fue la corrección del simulador. Para ello se comenzó por una fase de estudios en lo que refiere a protocolos y al *software* del propio simulador. Luego de esto se comenzó una fase de pruebas. En esta fase el grupo trabajó en conjunto, buscando detectar los errores de funcionamiento del simulador.

Según lo planificado se debería haber terminado la fase de detección para el 27 de julio de 2007. A esta fecha ya se habían detectado algunos errores. No obstante, como se verá, se encontraron más errores en instancias posteriores. Luego de este punto, se pasó a una fase de corrección y pruebas de verificación. También en esta fase se trabajó sin realizar división de tareas. Se optó por utilizar la metodología de programación extrema, dada la escasa experiencia en el área y que se estaba trabajando sobre un sistema desconocido.

Al principio del mes de Agosto se entró en un proceso iterativo e incremental, en el que se siguió hasta casi la finalización del proyecto. A medida que se fueron haciendo las pruebas de verificación de las correcciones realizadas, se fueron encontrando más errores. Estos eran corregidos y las pruebas revelaban nuevos errores y así sucesivamente.

En la segunda semana de Agosto, faltando un mes y medio para el primer punto de control de avance, se hizo un balance de lo que quedaba por realizar en esta etapa. Los errores encontrados tenían que ver con la implementación de dos protocolos, TCP y OSPF. En este punto se debió hacer una reestimación. Contando ya con un nivel de familiarización con el problema, se concluyó que sería imposible corregir todos los errores para la fecha inicialmente prevista. Por tal motivo fue necesario establecer prioridades.

Teniendo en cuenta que el criterio de éxito tenía que ver con el protocolo TCP, se decidió dar prioridad a la corrección de los errores relacionados con dicho protocolo. Desde ese punto, y hasta el fin de la primera etapa, el grupo se dedicó a corregir y documentar dichos errores. Cabe destacar que las pruebas realizadas sobre el protocolo TCP son independientes de OSPF.

Al fin de la primera etapa el criterio de éxito fue cumplido, el simulador corría adecuadamente la simulación preestablecida. Se tenían documentados todos los cambios realizados y se generó EasySim versión 2.1. Sin embargo, quedó pendiente la corrección de una serie de errores relacionados con la implementación de OSPF. Como se mencionó, el objetivo era que el simulador funcionará correctamente más allá del criterio de éxito. Con lo que, aún habiendo cumplido el criterio de éxito, se debería completar la corrección de los errores en la siguiente etapa del proyecto.

Naturalmente, el hecho de que el simulador no se encontrará totalmente corregido al fin de la primera etapa implicó una desviación en planificación. Este desvío es adjudicado a la gran cantidad de errores encontrados (por mayor detalle ver Anexo B). Cuando se realizó la planificación del proyecto no se esperaban encontrar tantos problemas en el simulador. Se tenía conocimiento de que habían desperfectos en la implementación del protocolo TCP (de ahí la elección del criterio de éxito). Sin embargo, los inconvenientes relacionados con OSPF no eran conocidos a priori. Estos fueron descubiertos en la fase de pruebas realizada en este proyecto. Se destaca además, que dichos errores eran de severidad alta, se interrumpía la ejecución de la simulación.

4.2.2 Segunda etapa

Esta etapa comienza a principio del mes de Octubre, en seguida del primer punto de control de avance. Como se mencionó en el apartado anterior, fue necesario apartarse un poco de la planificación. En vez de pasar a la implementación de las prácticas de laboratorio, como se había planificado inicialmente, fue necesario continuar con la corrección del simulador. Con lo que la primera tarea que se debió realizar en esta etapa fue una reestimación.

Buscando evitar un atraso mayor en la evolución del proyecto, se dividieron tareas. Al tiempo que dos integrantes estuvieron trabajando en la finalización de la etapa de correcciones, otro integrante procedió a comenzar la etapa relacionada con las prácticas de laboratorio. Como primer tarea para dicha etapa se había establecido el desarrollo del manual rápido para el simulador. Con lo que el responsable de esta tarea se dedicó a la implementación de dicho manual.

Simultáneamente, los demás integrantes se dedicaron a finalizar el trabajo sobre el simulador. De la misma manera que en la primera etapa, se siguió un proceso iterativo e incremental. Se profundizó en la evaluación del funcionamiento de OSPF, detectando y corrigiendo errores. Esta fase tuvo una duración aproximada de un mes. Punto en el cual se dio por finalizada la corrección de los errores del simulador, quedando documentados los cambios realizados y se generó EasySim versión 3.

A fines del mes de noviembre, cuando se finalizó la corrección de errores del simulador, se realizó un balance general de los avances que se tenían hasta el momento. En ese punto lo único que se tenía hecho de la etapa de las prácticas de laboratorio era el manual rápido. Se concluyó entonces que se tenía un retraso aproximado de un mes y medio. Fue necesario rever los tiempos de dedicación y la estimación de duración de las tareas que se realizarían de ahí en más. Las tareas fueron reasignadas en base a la nueva estimación.

El siguiente paso fue realizar la implementación del prototipo de laboratorio. Para esto se distribuyeron tareas como selección de materiales, configuración de la red que sería usada y familiarización con el generador de paquetes SCORM. Al tiempo que se fueron completando estás tareas, se fueron iniciando nuevas tareas como generación de los paquetes SCORM y definición de formato para las prácticas de laboratorio.

Una vez finalizada la implementación del prototipo se procedieron a realizar dos tareas. Por un lado se comenzó a gestionar la realización de una evaluación del mismo. Inicialmente se tenía como objetivo que dicho laboratorio fuera evaluado por docentes y estudiantes. Se hizo difícil obtener una evaluación por parte de estudiantes porque, debido a los retrasos, en este punto del calendario no se estaban dictando cursos (período de exámenes). Por tal motivo fue posible únicamente coordinar con docentes dicha evaluación. Por otro lado se migró el laboratorio de Performance de Redes que era implementado con EasySim1 a EasySim3.

Para el segundo punto de control de avance se llegó con el simulador corregido, el manual rápido implementado, una definición de formato tentativa[2] y dos laboratorios implementados. Según la planificación inicial en este punto ya se debería haber obtenido la evaluación del laboratorio y se deberían tener un total de cuatro prácticas implementadas. Con lo que el proyecto llegó retrasado a este punto.

El retraso de esta etapa fue atribuido, de la misma manera que en la primera etapa, al gran número de errores encontrados en el simulador. Se realizó un balance de actividades en este punto a fin de planificar la finalización del proyecto en tiempo y forma. En este se estimó que se necesitarían aproximadamente cuatro semanas para cumplir con las tareas que quedaron pendientes en la etapa.

La corrección del simulador realizada en la segunda etapa del proyecto tomó aproximadamente cinco semanas. De esto se concluyó que las medidas tomadas para paliar el retraso de la primera etapa lograron absorber una semana del retraso. Se podría pensar que las medidas no fueron suficientes. Sin embargo, se tuvo en cuenta que según la planificación inicial la tercera etapa sería dedicada solamente a generar documentación. O sea, esta tarea fue sobreestimada a propósito buscando dejar tiempo para solucionar retrasos. Además, las correcciones realizadas sobre el simulador ya se encontraban documentadas. Con lo que, se estimó que se había

[2] Se refiere a la definición de formato como “tentativa” porque podría ser modificada a causa de sugerencias obtenidas en la evaluación.

adelantado parte de la etapa de documentación y que sería posible cumplir con los tiempos acordados.

4.2.3 Tercera etapa

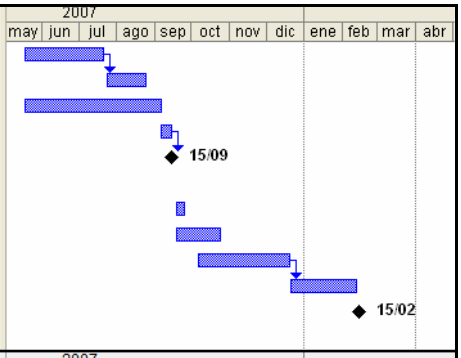
Según lo planificado inicialmente, en esta etapa se debería realizar toda la documentación del proyecto, diseñar un afiche para la muestra de proyectos y redactar un artículo en formato IEEE. No obstante, debido a los retrasos de las etapas anteriores, se debieron realizar las tareas pendientes de la segunda etapa.

Se comenzó por realizar las tareas pendientes dada su prioridad. Lo primero que se hizo en esta etapa fue obtener de la evaluación del prototipo por parte de docentes. Previo al comienzo de esta etapa ya se encontraba coordinada y preparada una presentación. De ésta se obtuvieron algunas sugerencias para mejoras en el laboratorio y principalmente en el simulador. Con lo que surgió una nueva tarea que fue la de implementar las sugerencias en el simulador. Además, esto conllevó una actualización en el manual rápido.

Las tareas fueron divididas de manera que se pudiera avanzar con la implementación de las prácticas de laboratorio en paralelo a la mejora del simulador. Una vez culminadas las tareas vinculadas al simulador, se comenzó la documentación. Cuando se terminaron las tareas vinculadas a los laboratorios, desarrollo y revisiones, se comenzó con la redacción del artículo en formato IEEE. Finalmente, se diseñó el afiche para la muestra de proyectos.

La figura 7 muestra un comparativo, en diagramas de Gantt, de la planificación del proyecto con el desarrollo real. Se pueden apreciar desviaciones, especialmente en lo que tuvo que ver con las correcciones del simulador. Éstas a su vez, generaron retrasos en las etapas posteriores. Gracias a la sobreestimación de la fase de documentación, fue posible cumplir con todos los objetivos del proyecto.

	Nombre de tarea	Duración	Comienzo	Fin
1	Detección	105 horas	mié 16/05/07	vie 20/07/07
2	Corrección	160 horas	lun 23/07/07	vie 24/08/07
3	Testeo	420 horas	mié 16/05/07	mié 05/09/07
4	Presentación p entregable 1	20 horas	mié 05/09/07	vie 14/09/07
5	**Entregable 1 Sim corregido**	0 días	sáb 15/09/07	sáb 15/09/07
6				
7	Manual rápido	15 horas	lun 17/09/07	lun 24/09/07
8	Prototipo	70 horas	lun 17/09/07	mié 24/10/07
9	Definición de formato	135 horas	vie 05/10/07	mié 19/12/07
10	Laboratorios	210 horas	jue 20/12/07	mar 12/02/08
11	**Entregable 2**	0 horas	vie 15/02/08	vie 15/02/08



	Nombre de tarea	Duración	Comienzo	Fin
1	Detección	105 horas	mar 22/05/07	jue 26/07/07
2	Corrección	233 horas	mié 01/08/07	vie 07/03/08
3	Testeo	521 horas	mié 16/05/07	vie 07/03/08
4	**Entregable 1**	0 días	sáb 15/09/07	sáb 15/09/07
5				
6	Manual rápido	42,5 horas	jue 08/11/07	vie 30/11/07
7	Prototipo	95 horas	lun 17/12/07	mar 12/02/08
8	Definición de formato	93 horas	mar 15/01/08	sáb 01/03/08
9	Laboratorios	110,5 horas	lun 21/01/08	lun 17/03/08
10	**Entregable 2**	0 horas	vie 15/02/08	vie 15/02/08

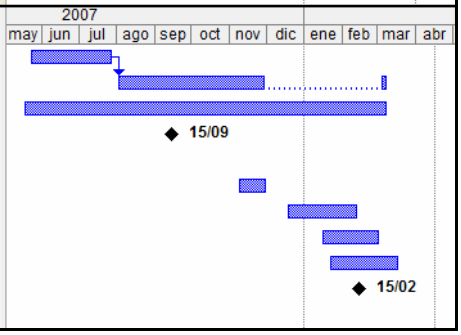


Figura 7: Comparativo del diagrama de Gantt estimado (arriba) y real (debajo)

Capítulo 5: Análisis y Diseño de la Solución

Este proyecto plantea un objetivo concreto, la implementación de prácticas de laboratorio virtuales. No obstante, como ya se mencionó, la resolución del problema planteado se separa en dos etapas. Por un lado se enfrenta el problema de corregir los errores del simulador y por el otro se afronta la implementación de las prácticas virtuales. Es por tal motivo, que también el análisis debe ser dividido en esas dos etapas. A continuación, se presentan los puntos claves de dicho análisis. Se detallan los problemas puntuales detectados y las soluciones propuestas.

La primera etapa del trabajo involucra solucionar los problemas del simulador EasySim2. Esto implica que se deba trabajar sobre un *software* de gran dimensión generado por otras personas. Es así que, previo a comenzar a trabajar sobre el código es necesario entender la arquitectura del programa y su funcionamiento. Luego se busca definir los problemas a resolver y su complejidad, tanto para el simulador como para la creación de los prácticas de laboratorio virtuales.

5.1 EasySim2

EasySim2 es un simulador de redes basado en el simulador de eventos discretos SimJava. EasySim2 está programado en lenguaje Java, utilizando la versión JavaTM 2 SDK, Standar Edition Versión 1.4.2. En el diseño de EasySim2, según dicen sus autores [2], se buscaron los siguientes objetivos:

- Extensibilidad: orientado a objetos para así facilitar el agregado de nuevas funcionalidades.
- De fácil uso: para ser utilizado por usuarios con conocimientos básicos de redes.
- Realista: que sea una buena herramienta, o sea, que permita tanto el estudio de redes en investigación como su utilización con fines didácticos.

Para tener una idea de las dimensiones del programa, se debe considerar que cuenta con 215 clases propias. A estas se les debe sumar las correspondientes a los paquetes del simulador de eventos discretos y paquetes de cálculo matemático, totalizando 348 clases.

5.1.1 Arquitectura

La figura 8 muestra la arquitectura del simulador en varios niveles. El primero divide el programa en dos bloques, EasySim y eduni (SimJava). En el segundo nivel, EasySim se divide en cuatro bloques principales: core, configuración, gui, e interfaz. A continuación analizamos cada bloque en forma individual.

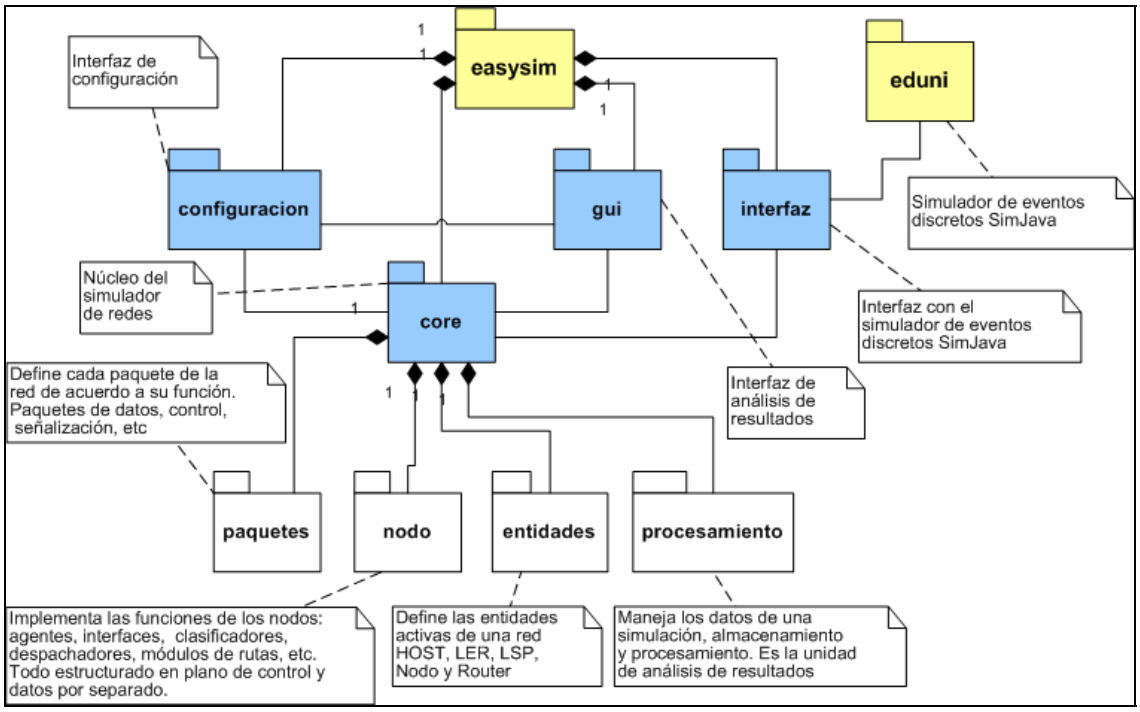


Figura 8: Esquema de Paquetes de EasySim2

Paquete Eduni – SimJava

En SimJava, cada sistema es un conjunto de entidades que interactúan entre sí. Estas entidades poseen puertos y se comunican una con otra enviando eventos a dichos puertos. Existe una clase (Sim_system) encargada de llevar el control de la simulación, mediante el control del avance del tiempo y del agendado y despacho de los eventos.

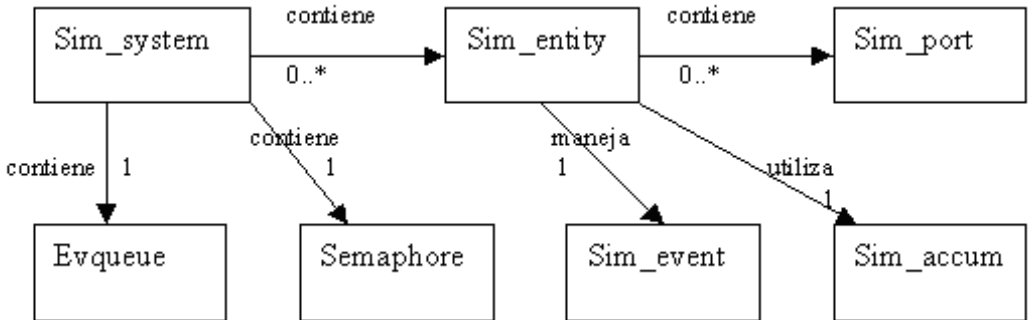


Figura 9: Diagrama de clases principales del paquete simJava de eduni

Sim_System: La clase Sim_System es la encargada de manejar la simulación. Una vez que la inicialización está completa, esta clase entra en el ciclo del método run(). Este corre todas las entidades (todos los hilos) en tanto éstas estén habilitadas, saca el siguiente evento de la *Future Event List* (FEL), lo procesa y repite el procedimiento.

Sim_entity: La clase Sim_entity es la clase de la cual deben heredar todas las entidades del sistema. Esta clase define el método body(), el cual debe ser sobrescrito por cada entidad que herede de Sim_entity para que cada una tenga el

comportamiento debido. Además, aquí se definen las operaciones que llevará a cabo una entidad. Dichas operaciones se pueden agrupar en cinco familias:

- métodos usados para agendar eventos.
- métodos usados para esperar eventos.
- métodos especializados en buscar eventos a ser procesados.
- métodos usados para procesar eventos.
- métodos usados cuando la entidad no está procesando ni esperando eventos.

Sim_event: Esta clase representa un evento de un sistema. Existen cuatro tipos de eventos:

- ENULL. Es el tipo por defecto que es asociado de inmediato, en el constructor de la clase cuando una entidad crea un evento para manejar los arribos que a ella llegan.
- SEND. Son creados por la clase Sim_System cuando se invoca el método send() de la misma, caso que ocurre cuando una entidad llama al método sim_schedule de la clase Sim_entity.
- HOLD_DONE. Es creado y agregado a la FEL cuando una entidad desea detenerse hasta un cierto momento de la simulación.
- CREATE. Es agregado a la FEL cuando se crea una nueva entidad en el sistema.

Sim_port: Esta clase representa un puerto mediante el cual una entidad se comunica con otra en el sistema. Cuando una entidad desea enviar un evento hacia otra entidad, procede enviando un evento hacia el puerto de la misma. Cada entidad puede tener más de un puerto, cada uno de ellos conectados a entidades distintas.

Evqueue: Esta clase representa la FEL del simulador de eventos discretos. Hereda de la clase Vector del *Application Programming Interface* (API) de Java y además, añade la funcionalidad de agregar los eventos en orden cronológico en la lista. Colocar los eventos en la FEL ordenados por sus tiempos de evento permite luego removerlos de manera rápida.

Semaphore: Esta clase es utilizada por la clase central Sim_System para sincronizar todas las entidades del sistema. Básicamente detiene y reinicia los distintos hilos de la simulación.

Por más detalles del funcionamiento de SimJava puede consultarse el Anexo C.

Paquetes de EasySim

A continuación se brinda una descripción de los bloques principales de EasySim2 basada en la documentación del programa [2].

Configuración

Este bloque se encarga de todo lo relacionado con la configuración de la red. El mismo administra tres elementos fundamentales: la configuración gráfica, la configuración de texto y el intérprete de texto. El intérprete es el encargado de realizar

la traducción de una configuración a otra, mientras el usuario esta diseñando una red. Las funciones principales son:

- Armado de la red, mediante la incorporación de los nodos, enlaces y creación de las interfaces.
- Configuración de los parámetros asociados a cada elemento de la red

A partir de la configuración realizada, el bloque core es el encargado de crear los elementos necesarios para realizar la simulación. Más información sobre el funcionamiento de este bloque puede encontrarse en [2].

Core

El bloque Core contiene todo lo concerniente a la simulación en sí. En él se simula el comportamiento de los distintos elementos que pueden componer una red.

- Host
- Links
- Routers

Los Hosts están compuestos por:

- agentes de tráfico.
- generadores de tráfico.
- un clasificador de paquetes.
- una interfaz.

Agentes de tráfico: Son los encargados de controlar tanto el tráfico proveniente de los generadores, como el entrante al Host. En EasySim2, existen 2 tipos de agentes, UDP y TCP. Este último en dos de sus versiones, TCP Reno y TCP Vegas.

Generadores de tráfico: Son los encargados de generar los diferentes paquetes de tráfico en la simulación. Cada agente de tráfico puede recibir los paquetes generados por uno o varios generadores de tráfico. En cada generador se puede elegir:

- los tiempos inter-paquetes, teniendo la opción de que sigan una determinada distribución probabilística o que sean determinísticos.
- los tamaños de los paquetes, teniendo la opción de que sigan una determinada distribución probabilística o que sean determinísticos.
- destino del paquete.
- clase de tráfico del paquete.
- prioridad al descarte del paquete.
- el tiempo de inicio y de término de generación de paquetes.

Clasificador: Es el elemento del Host que conecta a los agentes de tráfico con la interfaz del Host. Su función principal es la de enviar al agente debido los paquetes provenientes de la interfaz y viceversa.

Interfaz: Cumple dos funciones:

- recibe los paquetes provenientes del exterior del Host. Estos son recibidos y enviados directamente al clasificador del Host.
- recibe los paquetes provenientes del clasificador del Host. Según la política de descarte elegida para la cola de la interfaz, los paquetes son encolados. Luego despachados siguiendo la disciplina de despacho elegida para el despachador de la misma.

Cada interfaz tiene una cola física, la cual está dividida en n colas lógicas. A cada clase de tráfico, le corresponde una cola lógica.

Las políticas de descarte existentes en esta versión del simulador son:

- *Weighted Ramdon Early Detection* (WRED)
- *Drop Tail*

Las Disciplinas de Despacho disponibles son:

- Fifo
- *Deficit Weighted Round Robin* (DWRR)

Para que la interfaz tenga un comportamiento más realista, se le puede configurar una probabilidad de caída. De esta manera, durante la simulación, la interfaz puede caerse o levantarse con una probabilidad establecida por el usuario.

Los Routers están compuestos por:

- interfaces.
- módulos de control.
- modulo Ruteo de Capa de Red.
- modulo Unidad de Control.
- módulo Recursos.
- un clasificador de paquetes.
- un clasificador de paquetes de control.

Interfaces: Los Routers pueden tener más de una interfaz. Éstas están compuestas por una cola física con n colas lógicas en la cual los paquetes son encolados según una política de descarte determinada y un despachador que quita los paquetes de la cola siguiendo una disciplina de despacho.

Tiene dos funciones principales:

- enviar los paquetes que ingresan al Router al clasificador de paquetes.
- recibir paquetes de los distintos módulos del Router.

Los paquetes pueden ser de control o de datos. En caso que se trate de un paquete de datos, se intenta encolar. Si el paquete es de control, se envía directamente al Link sin pasar por la cola de la interfaz.

Clasificador de paquetes: La función del clasificador de paquetes es la de recibir los paquetes enviados desde la interfaz, identificar si se tratan de paquetes de datos o de control y enviarlos al módulo que realice el ruteo de los paquetes de datos o al clasificador de control respectivamente. En caso de que este activo el módulo

MPLS, se pasa el paquete al mismo, sino se pasa el paquete al módulo de Ruteo de Capa de Red.

Clasificador de paquetes de control: Los paquetes de control que recibe este clasificador son enviados al módulo de control adecuado.

Módulos de control: Cada Router puede tener distintos módulos que se encarguen del enrutamiento de los paquetes así como del descubrimiento y la actualización de la información de la red.

Módulo Ruteo de Capa de Red: Este módulo siempre está presente en los Routers. Es el encargado de dirigir los paquetes de datos provenientes del clasificador de paquetes cuando no se ha establecido el envío MPLS. La forma de enrutar los paquetes de este módulo está basada en el ruteo IP. Se busca el destino del paquete en una tabla de ruteo para saber por que interfaz se debe enviar el paquete.

Módulo Unidad de Control: Es el módulo encargado de avisar a los distintos módulos de control, que deben ejecutar las acciones debidamente agendadas.

Módulo Recursos: Es el módulo del Router que contiene toda la información que el Router conoce de la red a la cual pertenece.

Los Links son los elementos de la red encargados de unir los nodos de la red. Para modelar los enlaces reales, los enlaces a simular tienen:

- ancho de banda finito.
- probabilidad de caída.
- probabilidad de pérdida de un paquete.
- retardo.

Además de la definición de los elementos de la red, que se concentran dentro de los paquetes nodo y entidades (Ver Figura 8), el paquete core tiene otros dos paquetes importantes.

Paquetes: En éste se definen los paquetes válidos en la red. Pueden ser paquetes de datos, de control, de descubrimiento, de señalización, etc. Además de los paquetes que pueden circular por la red, se define un tipo de paquete que es interno al simulador PaqueteBasico y sirve para englobar todos los paquetes de TCP/IP. En total se definen 18 tipos de paquetes diferentes, incluyendo los necesarios para los algoritmos para OSPF y LDP (*Label Distribution Protocol*).

Procesamiento: Conjunto de clases que se agrupan en tres paquetes de acuerdo a la función que cumplen: datos, estadísticas y matemáticas. El paquete de datos es el que recopila y guarda la información durante la simulación. Además, es el encargado de comunicarse con los otros dos paquetes para presentar los datos al usuario. Los paquetes de estadística y matemática contienen clases que realizan los cálculos necesarios.

Gui

Es el paquete que brinda la cara visible del simulador. Aquí se define la estética de EasySim2. La interfaz gráfica está orientada a un entorno de ventanas, donde existe una encargada de dirigir las tareas de abrir y cerrar ventanas internas, llamada Ventana principal. Las ventanas internas se dividen en:

Configuración Texto: Se utiliza para la edición y alojamiento del texto de configuración.

Configuración Gráfica: Permite realizar la configuración de la simulación de manera visual y debe estar actualizada con respecto a la ventana de Configuración de Texto.

Animación: Muestra animaciones durante una simulación.

Tablas: Muestra las tablas de ruteo durante una simulación animada.

Resultados Gráficos: Permite visualizar, construir y guardar, una serie de gráficas a partir de los datos obtenidos de la simulación.

Resultados Estadísticos: Muestra estadísticas calculadas durante una simulación.

Interfaz

El paquete interfaz es el encargado de comunicar el simulador de eventos discretos SimJava con EasySim. Este bloque fue pensado para independizar el simulador de redes del simulador de eventos discretos que éste utiliza. En la Figura 10 se muestran: arriba las principales clases de SimJava y debajo las clases del paquete interfaz de EasySim.

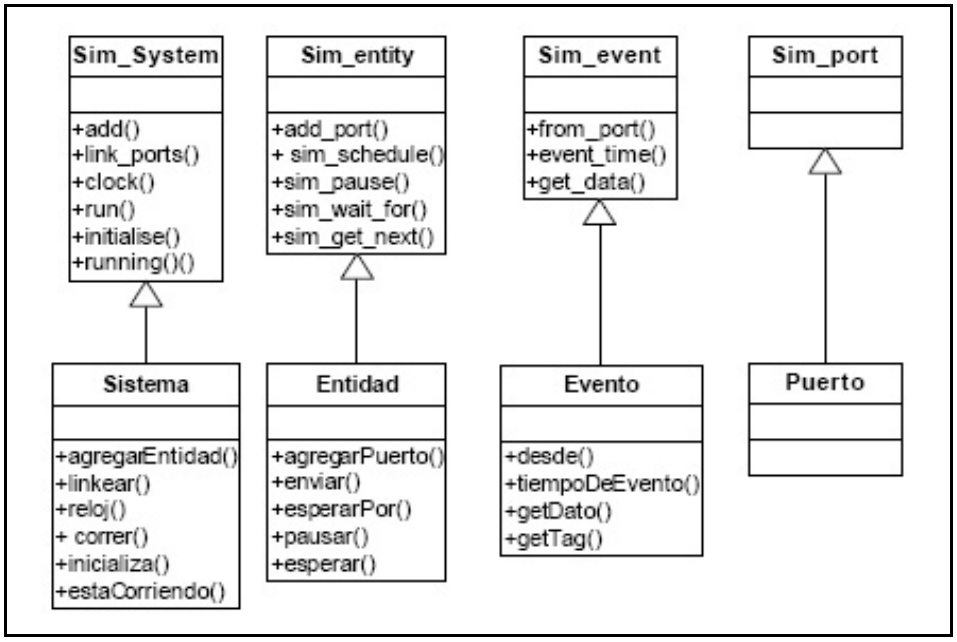


Figura 10: Interfaz de comunicación entre SimJava y EasySim (EasySim2-cap 2)

5.1.2 Hilos de ejecución

El simulador EasySim2 divide su funcionamiento en fases. La primera fase es la de configuración, en la que se genera la red y se crean las entidades necesarias para la simulación. Además, pueden agendarse algunos eventos que se ejecutarán durante la simulación. En esta fase las clases más relevantes son las del paquete configuración, junto a las del paquete nodo.

La segunda fase es la simulación, en ella el hilo de ejecución se realiza en base a eventos. Cada suceso que ocurre en la red representa algún evento, éste es recepcionado por una entidad y desata la acción correspondiente. Entre ellas, la

acción más común es el envío de paquetes de datos y de control. La clase fundamental en esta fase es *Sistema*. Ésta se encarga de llevar control del tiempo de simulación y de retirar los eventos de la cola de eventos.

La fase final es la de análisis de resultados obtenidos. Una vez terminada la simulación y recopilados los datos, éstos pueden ser examinados a través de las clases de procesamientos e interfaz de resultados.

Luego de esta reseña del funcionamiento, se analiza en detalle el hilo de ejecución de los protocolos TCP y OSPF.

Protocolo TCP

El protocolo TCP se estructura en torno a la clase *TCP_Connection*, ésta contiene todos los métodos que implementan una conexión TCP. En el mismo paquete se encuentra la clase abstracta TCP, de la cual heredan todos los agentes TCP. En la figura 11 se muestran las clases del paquete TCP, con excepción de *TCP_Packet* que es la clase de asociación.

TCP_Connection: Esta clase es la que maneja las conexiones. Se genera una instancia por cada conexión unilateral que exista en la red y se asocia a un agente TCP. La clase hereda de entidad, por tal motivo posee un *body()* el cual contiene un ciclo encargado de esperar y procesar los eventos.

TCP: Clase abstracta que define todos los métodos necesarios para simular el comportamiento del protocolo TCP. Se asocia a una instancia de la clase *TCP_Connection* mediante eventos de envío y la recepción de paquetes TCP (instancias de *TCP_Packet*).

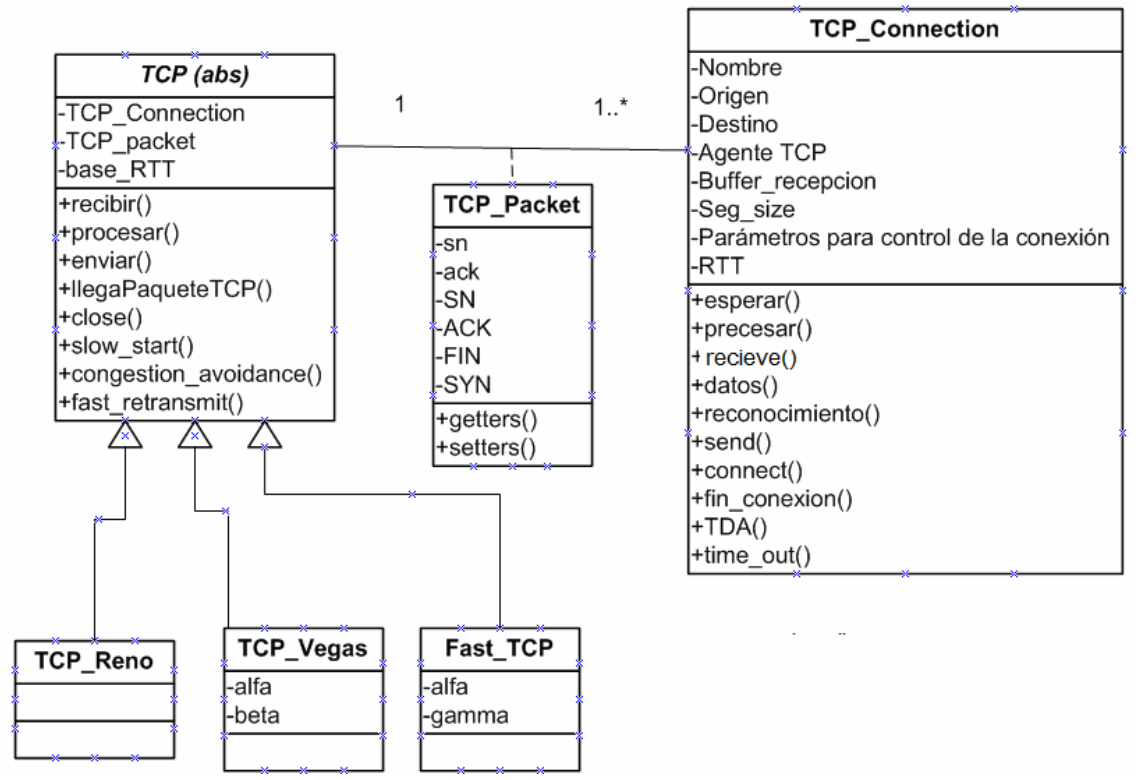


Figura 11: Diagrama de clases del paquete TCP (EasySim3)

TCP_Reno: Clase que implementa el protocolo TCP Reno para simular su comportamiento. Hereda de TCP.

TCP_Vegas: Clase que implementa el protocolo TCP Vegas para simular su comportamiento. Hereda de TCP e incorpora los parámetros específicos de este protocolo alfa y beta para realizar el control de congestión.

Fast_TCP: Clase que implementa el protocolo Fast TCP para simular su comportamiento. Hereda de TCP e incorpora los parámetros alfa y gamma para realizar el control de congestión. Esta clase no se encontraba en EasySim2

Protocolo OSPF

El protocolo OSPF es un protocolo de descubrimiento y se utiliza para mantener las tablas de ruteo actualizadas en redes dinámicas. El módulo de descubrimiento se configura en cada *router* de la red mediante tres parámetros:

HelloInteval: Define cada cuanto tiempo, en segundos, el *router* emitirá un mensaje Hello.

LSUInterval: Define cada cuanto tiempo, en segundos, el *router* emitirá un mensaje LSU.

K: Fija cuantos periodos de tiempo pueden pasar sin recibir un Hello de un vecino antes de considerarlo inalcanzable.

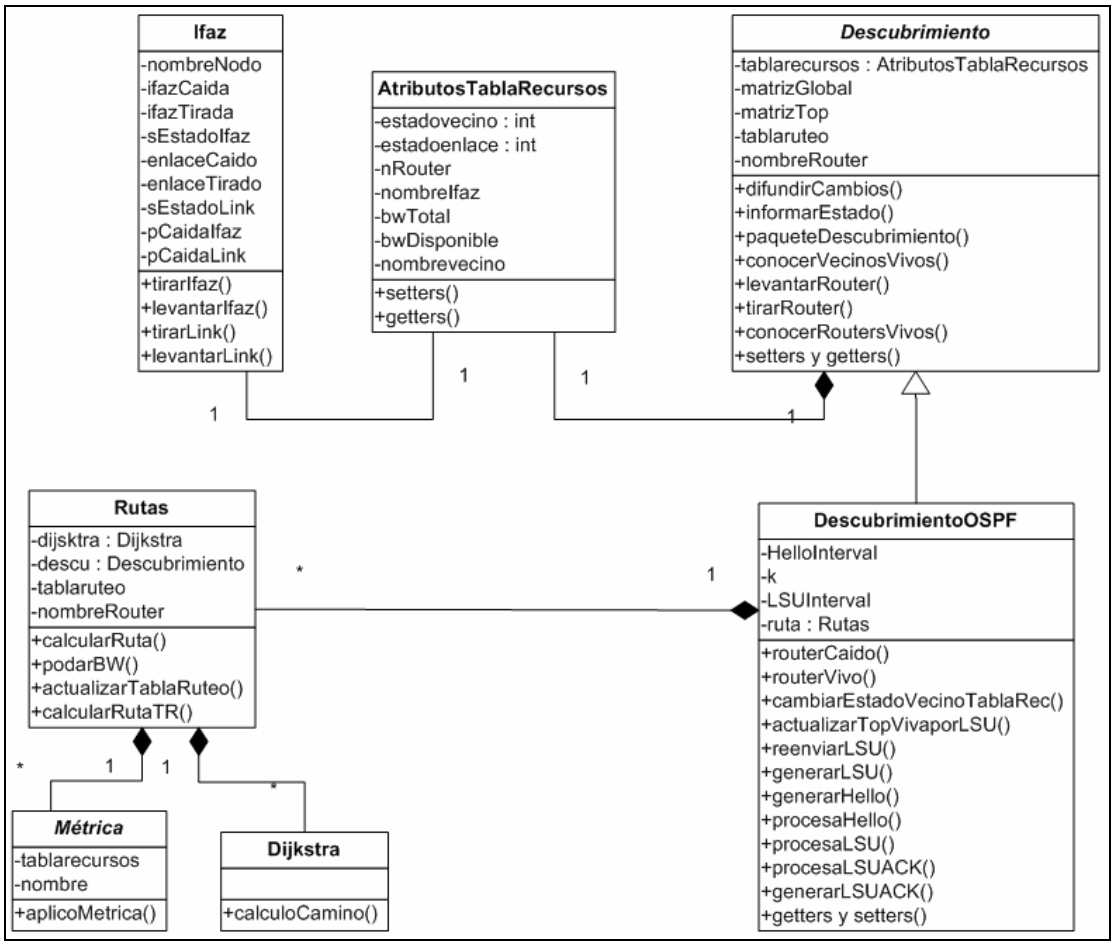


Figura 12: Diagrama de clases involucradas en la implementación de OSPF y ruteo dinámico

A continuación se detalla la cadena de eventos para determinar y actualizar el estado de una interfaz de un *router*. La matriz que se usa para calcular las rutas es la matriz *matrizTop*, la cual mantiene el estado de los enlaces que van del *router* origen (fila) al destino (columna); 1 = levantado, -1 = caído. Esta matriz se actualiza de acuerdo al estado que obtiene del parámetro *estadovecino* de la tabla de recursos del *router*. Dicha tabla y parámetro es modificado por el método *cambiarEstadoVecinoTablaRec* de la clase DescubrimientoOSPF. Este método utiliza el método *setEstadovecino* de la clase AtributosTablaRecursos. Además, al cambiar el estado se envía una actualización de estados por el método *difundirCambios.*

Dentro de la clase DescubrimientoOSPF tenemos una serie de métodos relacionado entre sí que son los encargados de simular las funciones de OSPF:

- difundirCambios
- generaLSU
- procesaLSU
- reenviarLSU
- generaACKLSU
- procesaACKLSU
- generaHello
- procesaHello

Los constructores de los paquetes *LSU*, *ACKLSU* y *Hello* están en sus respectivas clases dentro de easySim.core.paquetes.ospf.

Los eventos se identifican en el simulador por una etiqueta numérica. Cada evento refiere a una acción y esta acción está dicha en la etiqueta que lleva el evento. En el caso del protocolo OSPF se tienen las siguientes etiquetas:

30: informarestado

31: vecinocaido

32: reenviarLSU

33: DIFUNDIR_ESTADO

40: actualizarTablaRuteo

60: tirarIfaz

61: levantarIfaz

70: tirarLink

71: levantarLink

Asociado al funcionamiento de OSPF se tiene el módulo rutas. Cada vez que se actualiza la tabla de recursos, también se recalculan las rutas. La clase Rutas posee una instancia de Descubrimiento que le permite informarse de los cambios en la red. La clase Rutas posee los métodos necesarios para el cálculo y actualización de las tablas de ruteo asociadas a cada nodo. El cálculo es realizado utilizando la clase Dijkstra que implementa el algoritmo homónimo y aplicando la métrica correspondiente.

5.1.3 Uso de EasySim2

En esta sección no se busca dar un manual de uso del simulador, sino detallar las funcionalidades y las observaciones que se fueron realizando en la etapa de familiarización con EasySim2.

Las funcionalidades del simulador EasySim2 se dividen, de acuerdo a las capas del modelo *Open System Interconnection* (OSI), de la siguiente forma:

- A nivel de capa de Aplicación, posee diferentes generadores de tráfico.
- A nivel de capa de Transporte, se tienen los agentes: *User Datagram Protocol* (UDP) y *Transmission Control Protocol* (TCP). Este último en sus versiones TCP Reno y TCP Vegas.
- A nivel de capa de Red, se posee el enrutamiento basado en IP. Para el cálculo dinámico de las tablas de rutas, se tiene funcionalidades de *Open Shortest Path First* (OSPF).
- A nivel de capa Dos y Medio, se cuenta con algunas funcionalidades de *Multi Protocol Label Switching* (MPLS) y *Label Distribution Protocol* (LDP).
- Finalmente a nivel de capa física, se poseen las interfaces y los enlaces que conectan los distintos nodos de la red.

EasySim2 no requiere instalación, al descargar el programa se obtiene un archivo ejecutable de java (.jar) que puede ser abierto sobre cualquier JVM (*Java Virtual Machine*). Cabe mencionar que esta información no es brindada en el manual de usuario de programa. Al ejecutar el simulador nos encontramos con la ventana principal del mismo y lo primero a realizar es crear una nueva simulación.

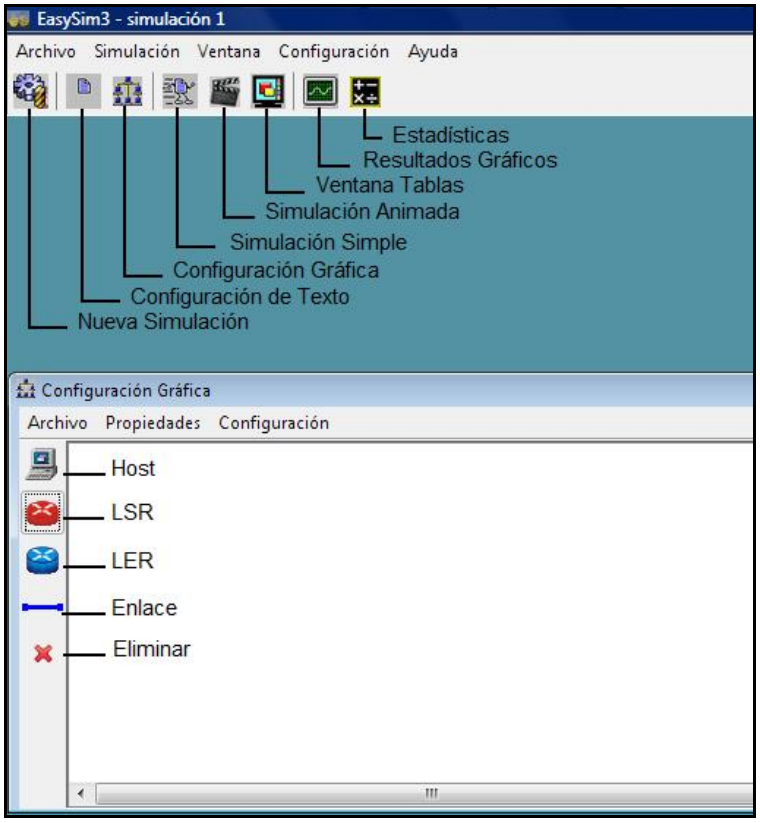


Figura 13: EasySim3 Ventana principal y Conf. Gráfica

En la Figura 13 se observa la ventana principal con fondo verde y conteniendo las barras de menúes y botones de acceso rápido. Debajo aparece la ventana de configuración en modo gráfico con fondo blanco y la barra de herramientas de diseño a la izquierda.

En este modo de configuración, basta con hacer un clic sobre cada elemento para agregarlos al área de diseño y luego arrastrarlo hasta su posición. Las interfaces se crean en forma automática al conectar un enlace a un nodo (Host, LER, LSR). Para fijar los valores de generadores, retardos en enlaces, anchos de banda, y el resto de los muchos parámetros disponibles en los elementos de la red, debe posicionarse el ratón sobre el elemento en cuestión y presionar el botón secundario. Así se despliegan en un cuadro a la derecha de la ventana cada parámetro que puede ser modificado. En este punto surge otro problema, no queda claro en el simulador las unidades que se utilizan para cada parámetro. Esta información sólo se brinda en un anexo del manual de usuario.

La interfaz gráfica es intuitiva y no es difícil llegar a diseñar una red simple, incluso sin haber leído todo el manual de usuario. Sin embargo, no es tan sencillo llegar a realizar una simulación exitosa. Si se diseña la red por entero en la interfaz gráfica y se quiere simular dicha red desde allí, la simulación no se efectúa. Siempre se debe correr la simulación desde la interfaz de texto, la cual se genera en paralelo, pero esto tampoco esta claro en la documentación. Otro detalle durante la simulación es que el reloj se muestra en la ventana principal (debajo a la derecha) y la ventana de simulación gráfica es independiente de ésta. Así, es conveniente ajustar estas ventanas en la pantalla de forma de ver la simulación y el reloj a la vez.

Finamente, después de simular una red, EasySim2 almacena una serie de datos. Éstos pueden ser consultados por el usuario a través de las ventanas de *Estadísticas* y *Resultados Gráficos*. Dichas funciones se encuentran muy bien documentadas, pero obtener un buen manejo de ellas lleva un tiempo razonable ya que son muy numerosas.

5.1.4 Diseño de la solución

Luego de la instancia de análisis del programa y bajo una serie de pruebas de funcionamiento se llegaron a detectar los siguientes problemas:

- Números de reconocimientos en TCP no son correctos
- Errores de funcionamiento del mecanismo de ventanas en TCP
- TCP carece de fin de conexión
- TCP Vegas tiene errores de implementación
- OSPF no actualiza correctamente las tablas de ruteo
- La interfaz gráfica carece de unidades

En lo referente a la solución de estos problemas, se propuso profundizar en el análisis de los algoritmos involucrados en cada uno de ellos. Con lo que, se generaron soluciones particulares para cada caso. Los lineamientos seguidos para dicho desarrollo fueron:

- No modificar más que lo estrictamente necesario, la interfaz gráfica del simulador
- Mantener la arquitectura

- Evaluar posibles mejoras en la interfaz con SimJava
- Realizar una batería de pruebas exhaustiva

Más información sobre la corrección del simulador puede encontrarse en el siguiente capítulo.

Desde el punto de vista de usuarios de EasySim2 se detectó que la documentación existente tiene algunas carencias y sobre todo es muy extensa. Por tal motivo uno de los primeros objetivos es mejorar esta documentación, complementando el manual de usuario existente con un manual rápido. El mismo debe tener menos de 10 carillas y ser basado en imágenes de forma que permita al estudiante en poco tiempo estar manejando las funciones básicas del simulador.

5.2 Prácticas de laboratorio

En los primeros dos capítulos se brindó el contexto en el que surge la necesidad de crear prácticas de laboratorio virtuales para el área de redes de datos. Los elementos principales se resumen a continuación:

- Masividad
- Escaso tiempo disponible para prácticas
- Elevados costos para mejorar los laboratorios físicos

El objetivo de estas prácticas de laboratorio es complementar las prácticas existentes y no sustituirlas. Sin embargo, buscan brindar al estudiante un aporte real de conocimiento, y motivarlo a experimentar más allá de lo visto en las clases regulares. Con éstos objetivos se plantearon los siguientes requerimientos:

- Accesibles desde web
- Auto-contenidos y reutilizables
- Autoevaluados
- Extender el campo de trabajo disponible en laboratorios físicos
- Formato predefinido
- Material teórico incluido

5.2.1 Análisis de requerimientos

Accesibles desde web, o en forma equivalente accesibles desde cualquier navegador web. Para ello el laboratorio debe componerse de uno o varios archivos que logren ser ejecutados desde un navegador. En casos puntuales pueden incluir pequeñas descargas, siempre que éstas estén justificadas y tengan instrucciones claras.

Considerando que el material de los cursos en el IIE se encuentra en Moodle [20], es razonable realizar las prácticas de laboratorio para que sean accesibles desde dicha plataforma. En este sentido existen dos posibilidades: armar las prácticas de laboratorio en Moodle con las herramientas que éste provee o realizar objetos SCORM que se pueden visualizar desde Moodle pero no se atan a la plataforma. En este sentido, no es difícil tomar una decisión: se opta por trabajar con objetos SCORM.

Auto-contenidos y reutilizables: Las prácticas de laboratorio se enfocan al área de redes de datos, pero no a un curso en particular. Por este motivo cada

laboratorio debe tratar un tema específico. En caso de ser necesario, se incluyen los requisitos previos para su realización. De ésta forma los laboratorios pueden utilizarse en varios cursos que toquen un mismo tema, sin excesivo trabajo por parte del docente.

Autoevaluados: Para comprobar que el estudiante ha adquirido o reafirmado un conocimiento importante, se debe realizar una evaluación. Si se considera las prácticas de laboratorio virtuales no necesariamente se realizan en presencia de un docente, la evaluación debe tener realimentación hacia el estudiante. En este sentido, trabajar con Moodle permitiría utilizar sus herramientas para el desarrollo de cuestionarios, sin embargo la elección de trabajar con SCORM obliga a buscar otra alternativa.

CreaQuiz [18] es un programa desarrollado por el Ing. Víctor González Barbone, que permite la generación de cuestionarios múltiple opción de respuesta automática. El mismo puede ser incorporado a un objeto SCORM y ejecutado en un navegador web, ya que el lenguaje utilizado en su desarrollo es JavaScipt. El programa presenta tablas conteniendo las preguntas y las opciones. Una vez que el estudiante finaliza el cuestionario, el programa retorna las respuestas correctas junto a la justificación de las mismas. Más información acerca de CreaQuiz se encuentra en la sección 5.2.2.

Extender el campo de trabajo disponible en laboratorios físicos: Los laboratorios ya existen, pero se encuentran limitados en su cantidad y variedad. Con la generación de prácticas de laboratorio virtuales, se debe mejorar en estos puntos. Por ejemplo, brindando mayor tiempo de práctica al estudiante, y mayor cantidad de situaciones de estudio (topologías).

Como ya se ha mencionado, para realizar prácticas de laboratorio en línea en el área redes de datos es imprescindible contar con un simulador. Éste es el que impondrá las limitaciones respecto a las topologías que se podrán estudiar, así como los temas que se podrán estudiar.

Formato predefinido: Este requerimiento apunta a la continuidad del trabajo realizado. La definición de un formato permite que las prácticas de laboratorio que se incluyen en MAPER y las que se generen en forma posterior mantengan coherencia entre sí. Además, facilita el desarrollo de nuevas prácticas de laboratorio.

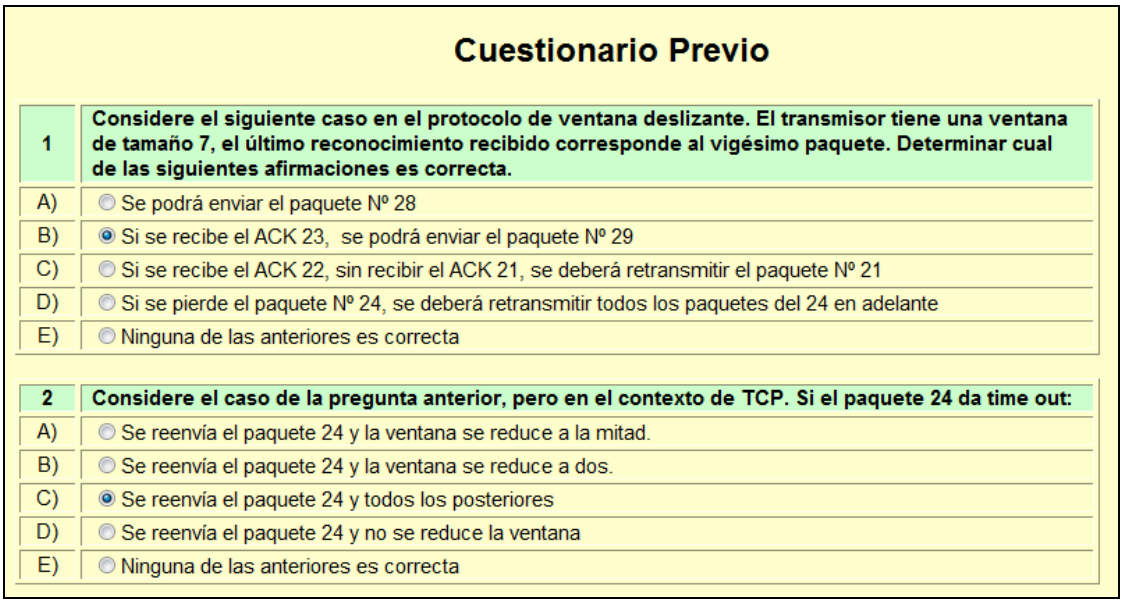
Material teórico incluido: Para que las prácticas de laboratorio sean realmente auto-contenidos, es razonable exigir que incluyan al menos un resumen de la base teórica del tema que tratan. Este puede ser brindado como parte del curso en que se incluya el laboratorio o independiente del mismo.

5.2.2 Herramientas utilizadas

CreaQuiz

Para la realización de los cuestionarios previos y posteriores se utiliza CreaQuiz [18]. Este programa permite realizar cuestionarios de múltiple opción en línea y brindar las respuestas al alumno en forma dinámica, mediante la generación de dos archivos HTML. Estos archivos son los que verá el alumno, uno con el cuestionario (Ver Figura 14) y otro con las respuestas (Ver Figura 15). Los mismos pueden diseñarse a gusto del docente, incluyendo tantas preguntas como se quiera y la cantidad de opciones que se ajuste a las necesidades. Simplemente debe

respetarse los formatos para definir las preguntas y las repuestas a fin de que el CreaQuiz pueda interpretarlas. Luego de que el estudiante contesta el cuestionario debe presionar el botón ***Enviar*** y obtendrá los resultados correspondientes.

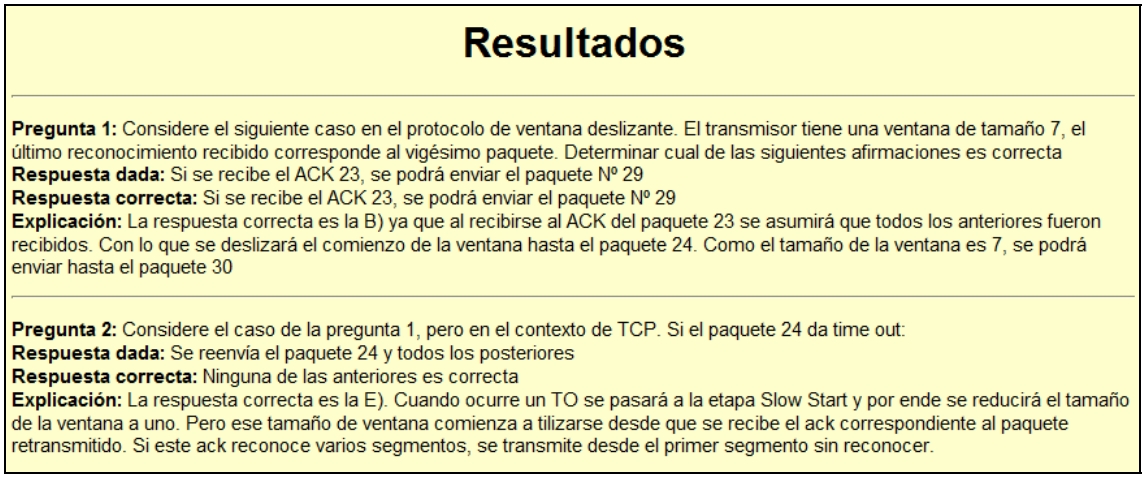


Cuestionario Previo

1	**Considere el siguiente caso en el protocolo de ventana deslizante. El transmisor tiene una ventana de tamaño 7, el último reconocimiento recibido corresponde al vigésimo paquete. Determinar cual de las siguientes afirmaciones es correcta.**
A)	○ Se podrá enviar el paquete Nº 28
B)	◉ Si se recibe el ACK 23, se podrá enviar el paquete Nº 29
C)	○ Si se recibe el ACK 22, sin recibir el ACK 21, se deberá retransmitir el paquete Nº 21
D)	○ Si se pierde el paquete Nº 24, se deberá retransmitir todos los paquetes del 24 en adelante
E)	○ Ninguna de las anteriores es correcta

2	**Considere el caso de la pregunta anterior, pero en el contexto de TCP. Si el paquete 24 da time out:**
A)	○ Se reenvía el paquete 24 y la ventana se reduce a la mitad.
B)	○ Se reenvía el paquete 24 y la ventana se reduce a dos.
C)	◉ Se reenvía el paquete 24 y todos los posteriores
D)	○ Se reenvía el paquete 24 y no se reduce la ventana
E)	○ Ninguna de las anteriores es correcta

Figura 14: Cuestionario con CreaQuiz



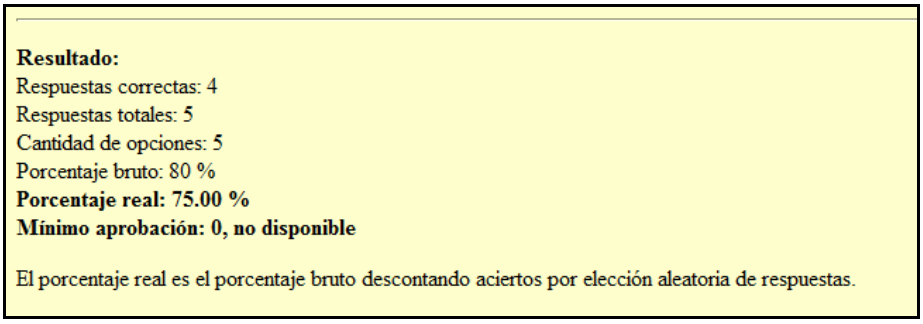
Resultados

Pregunta 1: Considere el siguiente caso en el protocolo de ventana deslizante. El transmisor tiene una ventana de tamaño 7, el último reconocimiento recibido corresponde al vigésimo paquete. Determinar cual de las siguientes afirmaciones es correcta
Respuesta dada: Si se recibe el ACK 23, se podrá enviar el paquete Nº 29
Respuesta correcta: Si se recibe el ACK 23, se podrá enviar el paquete Nº 29
Explicación: La respuesta correcta es la B) ya que al recibirse al ACK del paquete 23 se asumirá que todos los anteriores fueron recibidos. Con lo que se deslizará el comienzo de la ventana hasta el paquete 24. Como el tamaño de la ventana es 7, se podrá enviar hasta el paquete 30

Pregunta 2: Considere el caso de la pregunta 1, pero en el contexto de TCP. Si el paquete 24 da time out:
Respuesta dada: Se reenvía el paquete 24 y todos los posteriores
Respuesta correcta: Ninguna de las anteriores es correcta
Explicación: La respuesta correcta es la E). Cuando ocurre un TO se pasará a la etapa Slow Start y por ende se reducirá el tamaño de la ventana a uno. Pero ese tamaño de ventana comienza a tilizarse desde que se recibe el ack correspondiente al paquete retransmitido. Si este ack reconoce varios segmentos, se transmite desde el primer segmento sin reconocer.

Figura 15: Respuestas del cuestionario con CreaQuiz

Al final de documento de resultados se muestra el resultado total del cuestionario. Se brindan dos totales en porcentajes. El porcentaje bruto corresponde a la cantidad de aciertos sobre preguntas, mientras que al real se le deduce la probabilidad correspondiente a los aciertos aleatorios.



Resultado:
Respuestas correctas: 4
Respuestas totales: 5
Cantidad de opciones: 5
Porcentaje bruto: 80 %
Porcentaje real: 75.00 %
Mínimo aprobación: 0, no disponible

El porcentaje real es el porcentaje bruto descontando aciertos por elección aleatoria de respuestas.

Figura 16: Resumen de respuestas

RELOAD

Generar paquetes SCORM de acuerdo al estándar requiere la utilización de un editor con el fin de facilitar la inclusión de los metadatos en los objetos de aprendizaje. El programa elegido a estos efectos es RELOAD (*Reusable E-Learning Object Authoring & Delivery*). Este programa es fácil de utilizar, es liviano, de fácil instalación, y no requiere licencia (*software* libre) [16] [26].

El editor RELOAD se divide en tres paneles fundamentales (ver Figura 17). A la izquierda se encuentran todos los archivos que pueden ser utilizados dentro del SCORM (contenedor). Para ello se disponen de funciones para importar información y eliminarla del contenedor. A la derecha y arriba tenemos dos árboles; el superior corresponde al índice (Organización) del SCORM, o sea, las secciones que aparecerán y en el orden que aparecerán. El árbol inferior lista los recursos (Recursos). Cada link del primer árbol puede corresponderse a uno o más archivos del contenedor, pero sólo a un recurso. La asociación se hace entre archivos del contenedor y un recurso. Luego este recurso es asociado a un ítem de la Organización. Finalmente, a la derecha debajo se encuentran los campos para generar las asociaciones y definir la nomenclatura a utilizar.

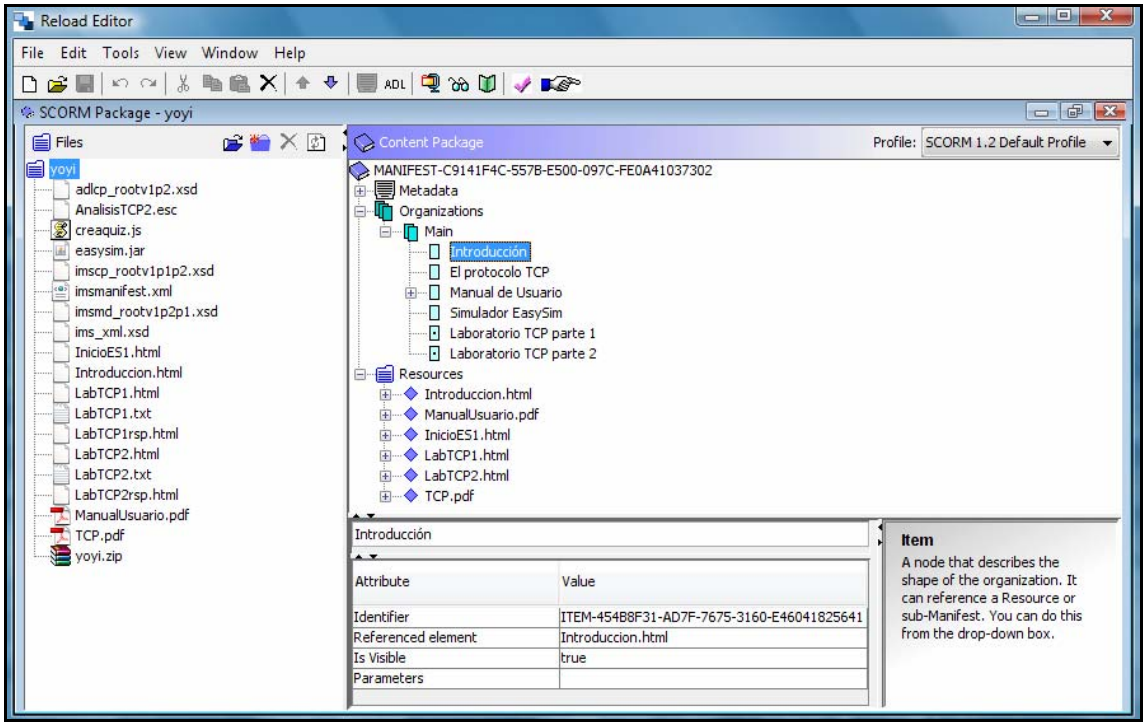


Figura 17: Editor RELOAD

Moodle

En el capítulo 3 se describen las características de los LCMS y en particular de Moodle. Siendo esta la plataforma que se encuentra en uso actualmente en el instituto, resulta necesario que las prácticas de laboratorio diseñados se testearan en la misma. Por tal motivo, realizamos la instalación local de dicho sistema.

Moodle es una plataforma de educación que requiere ser instalada sobe un servidor web y relacionada a una base de datos. Hace algunos años era necesario instalar cada programa de tal forma que pudieran comunicarse, pero ahora se dispone de paquetes que realizan toda la instalación y configuración en pocos pasos simples [22]. Una vez instalado Moodle, se ingresa al entorno de trabajo mediante el navegador web accediendo al *localhost*[3].

5.2.3 Diseño de la solución

El problema a ser resuelto fue realizar cuatro prácticas de laboratorio, sujetas a los requerimientos ya descritos. A tal fin, se propuso desarrollar un proceso incremental. El mismo se compuso de las siguientes etapas:

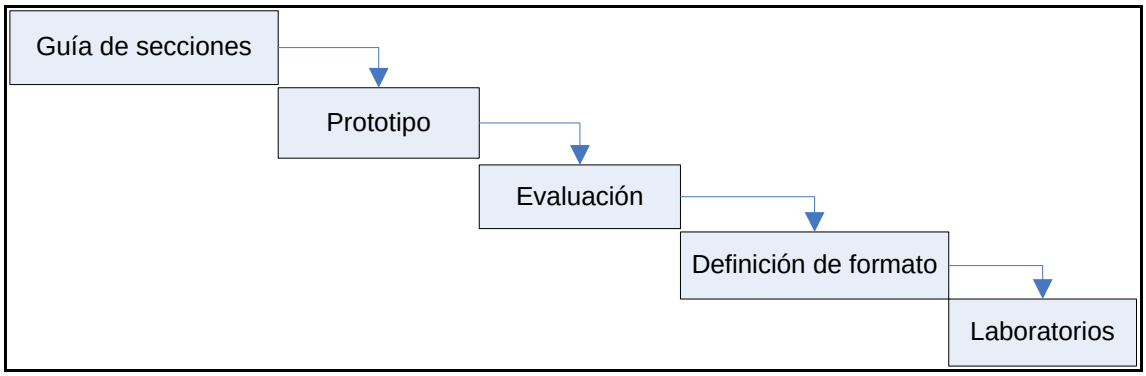


Figura 18: Proceso de diseño de las prácticas de laboratorio

Guía de secciones: Se propuso una definición tentativa de las secciones que componen el laboratorio. Esta definición serviría de guía para desarrollar el prototipo y podría ser modificada luego de la evaluación. Las secciones propuestas en esta primera instancia fueron:

1. Introducción
2. Breve desarrollo teórico
3. Bibliografía recomendada
4. Ejercicios preparatorios para la realización del laboratorio
5. Solución de los ejercicios pre-laboratorio
6. Planteo del laboratorio
7. Ejecución del laboratorio (Por ejemplo: cargar el Simulador)
8. Ejercicios pos laboratorio y/o evaluación del mismo
9. Solución de los problemas planteados en el punto anterior
10. Devolución al estudiante sobre su desempeño en la práctica

Prototipo y Evaluación: Se propuso implementar un prototipo de laboratorio siguiendo las secciones antes mencionadas. Una vez desarrollado, éste debería ser evaluado por personas externas al proyecto como ser docentes y/o alumnos de cursos

[3] Si se utiliza Mozilla Firefox debe digitarse la dirección IP correspondiente al *localhost*.

relacionados. El objetivo de dicha evaluación sería mejorar el prototipo y en base a eso implementar los siguientes laboratorios.

Definición de formato: Partiendo del prototipo y de la evaluación, se propuso realizar una definición de formato. Ésta no sería una regla estricta a ser seguida en la implementación de futuros laboratorios. Su finalidad sería buscar facilitar trabajos futuros y asegurar la coherencia de los mismos.

Laboratorios: Una vez finalizada la definición de formato y con una práctica de laboratorio implementada, sería necesario implementar tres laboratorios más para cumplir con el objetivo. Para esto se propuso que cada uno de los integrantes desarrollara una práctica.

En el siguiente capítulo se presentan los detalles de cómo se implementó este proceso. Se explican además, algunos puntos en los que fue necesario desviarse del camino planificado.

Capítulo 6: Desarrollo del proyecto

Este capítulo presenta el desarrollo del proyecto. Se profundiza en la explicación brindada anteriormente acerca de las soluciones propuestas, brindando referencias a anexos donde se puede encontrar la información completa. Se describen además, las opciones manejadas en las diferentes etapas.

De la misma manera que en los capítulos anteriores, el presente divide su enfoque en dos secciones. Por un lado se presentan las implementaciones de las soluciones propuestas para la corrección del simulador. Mientras que por el otro, se describe el desarrollo de las etapas vinculadas con las prácticas de laboratorio.

6.1 Corrección de EasySim

El análisis presentado en el capítulo anterior devolvió una serie de problemas con el simulador que deberían ser corregidos (ver apartado 5.1.4). Esta sección describe con más detalle los errores encontrados y como fueron resueltos. Se asume aquí que el lector tiene conocimientos básicos de redes de datos (TCP y OSPF esencialmente), condición necesaria para comprender las soluciones propuestas.

El proceso seguido fue iterativo e incremental. A medida que se fueron corrigiendo errores se realizaron pruebas para verificar el correcto funcionamiento. A su vez, estas pruebas revelaron nuevos errores, y así sucesivamente. Buscando facilitar la comprensión, esta sección es dividida en sub secciones que abordan por separado los diferentes errores. Asimismo, se mencionan las pruebas realizadas de manera que se pueda tener una idea cronológica de los acontecimientos. Los detalles de la evolución cronológica y de los cambios realizados sobre el código pueden ser encontrados en el Anexo B.

6.1.1 Problema con ACK

Este problema fue detectado en la etapa de familiarización con el simulador. Se realizó una simulación muy simple, un *host* transmitiendo y otro recibiendo datos. En esta prueba se pudo observar un error en el número de reconocimiento. La RFC 793 [27] establece que el valor del ack enviado debe ser el número del próximo byte que se espera recibir. El simulador, hasta este punto del proyecto, manejaba números de secuencia y reconocimientos en paquetes y no en bytes. Por lo tanto, el ack que debía ser enviado correspondería al próximo segmento esperado. En lugar de esto, se estaba enviando el valor del último segmento recibido.

```
14.1348 r: TCP: size: 1500.0(bytes) sn: 9 ack: 0 ACK: 0
14.1348 t: TCP: size: 20.0(bytes) sn: 0 ack: 9 ACK: 1

14.1348 r: TCP: size: 1500.0(bytes) sn: 9 ack: 0 ACK: 0
14.1348 t: TCP: size: 20.0(bytes) sn: 0 ack: 10 ACK: 1
```

Figura 19: Ack en TCP, antes (arriba) y después de corregir (debajo)

Dicho error fue corregido mediante modificaciones en el método datos() de la clase TCP_Connection. Una vez realizadas las correcciones se procedió a la

realización de pruebas de verificación. En éstas se observó un comportamiento acorde a la definición del protocolo.

En etapas de corrección posteriores se detectaron problemas con el número de segmento que se estaba enviando al ocurrir una pérdida. Un análisis de dichos comportamientos evidenció que la solución propuesta para el problema del ack no había sido contemplada en todos los algoritmos que se veían afectados. Una vez realizadas las correcciones pertinentes, se procedió a efectuar una revisión profunda a fin de evitar la aparición de nuevos problemas.

6.1.2 Mecanismo de ventanas en TCP

En lo que refiere a este tema, varios errores fueron encontrados. Los primeros fueron detectados en las pruebas mencionadas en el apartado anterior. Ya en estas se observaba que el establecimiento de la conexión no se daba en forma inmediata. Una serie de pruebas evidenció una relación entre dicho retraso y el tiempo inter-paquete[4]. Se esperaba que el inicio de la conexión se diera luego de generado el primer segmento de datos. En cambio, se observó un retraso, directamente proporcional al tiempo inter-paquete, entre el inicio de la generación de datos y el envío del primer segmento TCP (SYN=1).

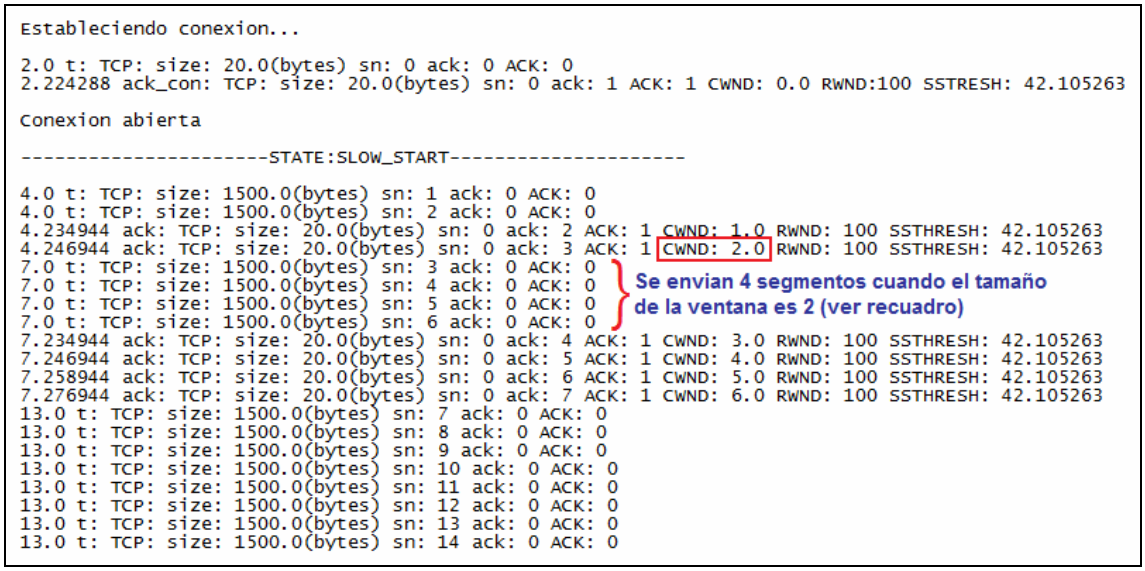


Figura 20: Archivo de salida de TCP, con problemas en la ventana de transmisión

Esto motivó un estudio de las clases encargadas de la generación de datos. Se detectó que el algoritmo de generación de datos tenía un error. Más precisamente, el problema estaba en las llamadas a pausas del generador de eventos discretos. Éstas eran realizadas antes de lo debido, con lo que se esperaba un tiempo inter-paquete antes del inicio de la conexión.

Buscando verificar las correcciones, se realizaron más pruebas. Se eligió como simulación en este caso, la de la primera práctica de la asignatura Evaluación de Performance en Redes de Datos. La elección de esta configuración se debió a que el

[4] El simulador brinda la posibilidad de configurar dos parámetros en lo que tiene que ver con los datos generados a nivel de aplicación. Uno es el tamaño de los “paquetes” que se generan. El otro es el tiempo entre paquetes, tiempo inter-paquete.

criterio de éxito elegido fue un buen funcionamiento del simulador ante esta simulación. Esta prueba arrojó una serie de errores relacionados con el protocolo TCP.

El primer problema que se abordó fue una inconsistencia entre los segmentos que se enviaban y el tamaño de la ventana de transmisión. Los segmentos eran enviados según el tamaño que debería tener la ventana. En cambio, el valor de la misma era actualizado recién después de la recepción del primer reconocimiento de un segmento de dicha ventana. En conclusión, el programa enviaba correctamente los segmentos pero no actualizaba los archivos de salida en el momento correcto. Este problema fue resuelto modificando las llamadas a actualizaciones de los archivos de salida.

El error mencionado en el párrafo anterior no fue el único detectado con esa prueba. Se observó también, que una vez recibido un reconocimiento (ACK), no se enviaban más segmentos en forma inmediata. Según lo que se podía ver en los archivos de salida, la cantidad de segmentos enviados era siempre igual a la capacidad de la ventana de transmisión. Con lo que se concluyó que el simulador realizaba envíos solamente cuando se tenía datos suficientes para llenar dicha ventana.

Un posterior análisis del código reveló problemas de implementación de algoritmos. Entre estos se destacan errores en la actualización de un contador y en las condiciones de salida de un ciclo. Estos generaban funcionamientos anómalos como ser envío de segmentos sin que existieran datos y no envío de segmentos hasta que se creara una cantidad de datos muy grande. El Anexo B explica claramente esta situación y las medidas tomadas para resolverla.

Una vez resueltos estos problemas, se descubrió un caso específico que no era contemplado por la implementación del simulador. Éste ocurría cuando la cantidad de datos (de capa 5) que se tenían en el buffer de TCP no era suficiente para completar una MTU[5]. Se encontró que un segmento era enviado solamente cuando se tenían datos suficientes para completar una MTU. Con lo que, cuando el generador de datos (capa 5) tenía una tasa muy lenta, los datos deberían esperar un tiempo muy largo para que fueran enviados. Incluso, si no se generan más datos, nunca serían enviados los que estaban en el *buffer*. Para evitar esto se implemento un envío por *time-out*, cuya función es precisamente limitar el tiempo de espera de los datos.

En las primeras pruebas realizadas sobre esta implementación se detectaron ciertas carencias. La primera versión del envío por *time-out* no contemplaba el tamaño de la ventana de transmisión. Una versión mejorada de esta implementación fue revisada mediante una serie de pruebas. En todas estas se realizaron simulaciones con baja tasa de generación de datos. Se obtuvieron resultados satisfactorios: los envíos se realizan contemplando el tamaño de la ventana, ya no se apreciaron demoras en el envío de los segmentos y se enviaron segmentos de tamaño variado (recordar que sólo se estaban enviando segmentos de tamaño MTU).

[5] MTU: Maximum Transfer Unit, mayor carga útil que puede tener un segmento TCP.

6.1.3 Fin de la simulación

Durante la realización de las pruebas antes mencionadas, y prestando especial atención al fin de las simulaciones, se encontraron algunos errores. Más precisamente, se encontraron dos comportamientos anómalos. El primero, fue que el simulador no tenía implementado el fin de conexión, recordar que TCP define explícitamente una etapa de liberación de la conexión. Teniendo en cuenta que el simulador tiene un fin pedagógico, se implementó el fin de conexión. Para esto, se debieron modificar los métodos que se encargan de enviar y recibir los segmentos. Además, fue necesario realizar algunas modificaciones en el segmento de fin de conexión, para que fuera identificado como tal.

El segundo problema, fue detectado en una simulación en la que se configuró un tiempo de apagado para el generador de datos (capa 5) menor al tiempo de simulación. A priori, Se esperaría que la simulación continuara hasta el tiempo preestablecido (duración de la simulación). En cambio, ésta se detenía en el tiempo de apagado del generador. Con lo que era un error de severidad alta. Cabe destacar que si existían datos por ser enviados en el buffer en el momento en que se apagaba el generador, estos no eran enviados.

Lo primero que se supuso fue que, dado que los tiempos son manejados por un simulador de eventos discretos, la simulación se cortaba porque no existían más eventos agendados. Se realizaron pruebas buscando corroborar esta suposición. Se configuró una simulación en la que, posterior al tiempo de apagado del simulador, debería caerse un enlace. Como se presumió, la simulación avanzó hasta el instante en que se agendó la caída del enlace.

Teniendo reforzada la suposición, se propuso resolver el problema agendando un evento de fin de simulación (por detalles del manejo de eventos ver sección 5.1.1). Para esto, sería necesario que alguna entidad levantara ese evento. Buscando evitar modificar las demás clases (lineamiento propuesto inicialmente, modificar lo mínimo posible) y teniendo en cuenta que se estaba agregando algo nuevo, se decidió crear una nueva clase. Ésta se llamó Fin y tuvo la propiedad de heredar de la clase Entidad, condición necesaria para que pudiera levantar eventos.

Luego, para completar la solución del problema, faltaba elegir el método desde donde se lanzaría el evento fin. Lo que en principio pareció una tarea simple reveló sus dificultades. En varios lugares donde se probó (empíricamente) se obtuvieron errores en tiempo de ejecución porque estos no podían lanzar eventos.

Finalmente, se encontró que la clase GeneradorBásico tenía alcance suficiente como para que se agendara el evento fin de simulación. Durante este último análisis, se descubrió que no era necesario levantar el evento en otra entidad. Pruebas realizadas comprobaron que era suficiente con agendar un evento para el fin de la simulación para que ésta llegara a su final. Teniendo en cuenta esto y los lineamientos propuestos (realizar la menor cantidad de cambios posibles al código) se decidió eliminar la clase Fin. Una vez finalizadas estas correcciones se verificó el correcto funcionamiento mediante una serie de pruebas.

6.1.4 TCP Vegas

Durante la realización del laboratorio de ventanas se detectó que la implementación de TCP Vegas no era consistente con lo establecido en el protocolo. En la ejecución se observaba un pasaje por la etapa *Slow Start* (crecimiento de la ventana en forma exponencial) que no existe en la definición.

El mecanismo de ajuste de ventana seguido por esta versión de TCP es el siguiente: se aumentará el tamaño de la ventana de transmisión en uno cada vez que se reciba el reconocimiento de un segmento en un tiempo menor a un determinado umbral. Se corrigió la clase TCP_Vegas de forma que desde el principio comience realizando el algoritmo correspondiente.

En la siguiente figura, se puede observar la diferencia de comportamiento del tamaño de las ventanas, a la izquierda antes de corregir y a la derecha luego de quitarle el *Slow Start*:

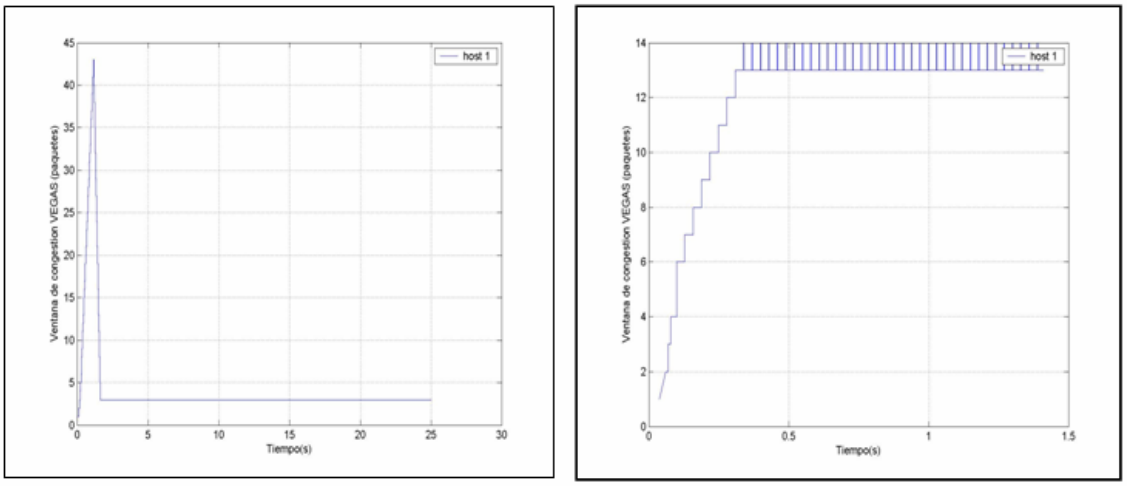


Figura 21: Comparación del tamaño de la ventana de congestión de TCP_Vegas

Otro problema que se detectó luego de solucionar el comportamiento de TCP_Vegas fue que al imprimir los archivos de salida se seguía mostrando una etapa *Slow Start*. Se debió modificar la clase TCP_Connection de manera que discrimine si la conexión tiene agente TCP_Vegas o TCP_Reno, de forma de imprimir la etapa que corresponda.

6.1.5 OSPF

En lo que refiere a este protocolo, se encontraron errores ya durante la etapa de familiarización con el simulador. Como en esa etapa también se habían encontrado problemas con la implementación de TCP (los descritos anteriormente), fue necesario establecer prioridades. Se decidió entonces comenzar con la corrección de TCP porque el criterio de éxito elegido está vinculado a éste. Con lo que las correcciones referidas a OSPF fueron realizadas en instancias posteriores.

Al tratarse OSPF de un protocolo de ruteo dinámico, para probar su correcto funcionamiento se simuló una red cuya topología cambiara durante la simulación. Lo que se propuso fue agendar la caída de un enlace y verificar que ese cambio de

topología se reflejara en las tablas de ruteo. Lo que ocurrió fue que la simulación se cortó en el momento en que se cayó el enlace. Con lo que también aquí, se encontró un error de severidad alta.

Se comenzó entonces por resolver el problema de la interrupción de la simulación. Al profundizar en el análisis de esta parte del código, se encontró un error en el manejo de una excepción. Se observó que un método lanzaba un tipo de excepción que no era tratada. La solución implementada para este problema fue crear un bloque *try-catch* que trata el eventual lanzamiento de la excepción mencionada.

Mediante pruebas se pudo comprobar que la solución propuesta evitaba que se cortara la simulación. Sin embargo, al caer una interfaz o enlace no se actualizaban las tablas de ruteo. Se continuó entonces con la revisión de los algoritmos involucrados en la implementación del protocolo OSPF. En ésta, se encontró un error en el uso de un parámetro. Se modificó el método de modo que se usara el parámetro correcto.

Esto solucionó el problema de las actualizaciones en las tablas de ruteo y con ello se verificó que OSPF funcionaba en forma satisfactoria. Reconoce los cambios en la red y se ajusta a ellos. Además, cabe destacar que este cambio también eliminó la excepción que se generaba, dejando de existir la necesidad del bloque *try-catch*.

6.1.6 Modificaciones en la interfaz gráfica

Inicialmente, no se pretendían realizar cambios en la interfaz gráfica ya que esta es bastante amigable y se buscaban hacer la menor cantidad de cambios posible. No obstante, se encontró un punto en el cual se podrían hacer mejoras. Las tablas de configuración de parámetros no contaban con unidades. Éstas se encuentran descritas en el anexo del manual original, con lo que no son fácilmente accesibles.

Teniendo en cuenta lo anterior, se decidió modificar la interfaz gráfica agregándole las correspondientes unidades. La Figura 22 muestra una comparación de la interfaz antes y después de los cambios. Se puede apreciar claramente las unidades en los respectivos parámetros. Asimismo, se destaca que este fue el único cambio que se realizó en la interfaz.

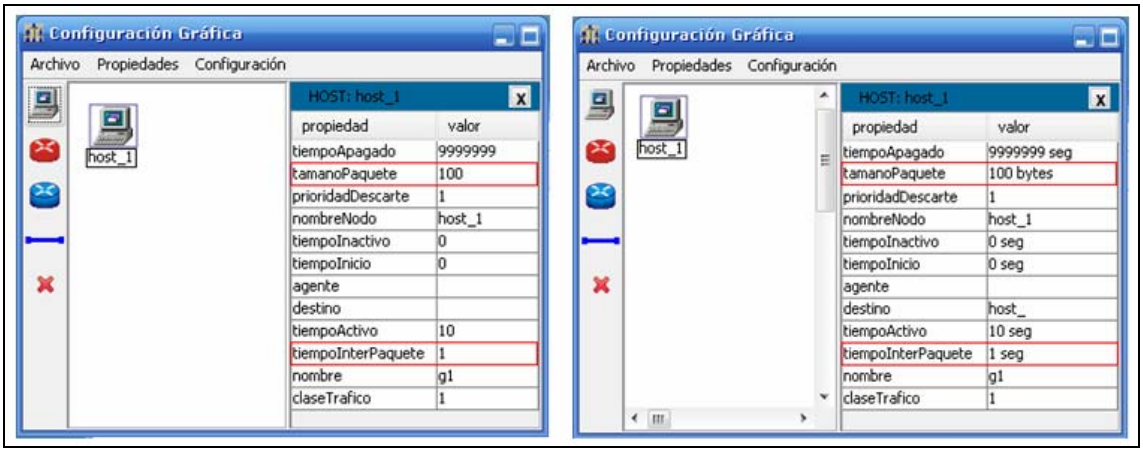


Figura 22: Evolución de la interfaz gráfica

6.1.7 Observaciones sobre EasySim3

De las pruebas se desprendió una carencia del simulador, no es posible implementar un tráfico bi direccional como en la realidad. La única forma de hacerlo es iniciando dos conexiones unidireccionales independientes. Aún así, las conexiones no deben iniciarse simultáneamente porque de ser así se interrumpe la simulación.

Si un enlace es tirado desde uno sólo de sus extremos la simulación no se corta, pero se pierden todos los paquetes mientras este caído el enlace y los paquetes no se re-enrutan. Para usar la función TirarLink deben tirarse ambos extremos del enlace.

Si los paquetes generados en capa 5 son muy pequeños, con tiempos de generación altos (poco tráfico) y se usa el protocolo TCP puede surgir un problema de transmisión. Este problema es que se envía únicamente el primer paquete de la conexión y nunca se envían los restantes. Este es un caso muy particular que no es representativo del comportamiento usual de una red.

Al tirar al menos una interfaz correspondiente a un enlace, se provoca la caída de dicho enlace.

Se dejó documentado estos comportamientos, ya sea para ajustes en una futura versión o simplemente para conocimiento.

6.2 Prácticas de Laboratorio

Recordando, el proceso propuesto en la sección 5.2.3 del presente informe para el diseño de las prácticas de laboratorios era:

$$\text{Guía de secciones} \Rightarrow \text{Prototipo} \Rightarrow \text{Evaluación} \Rightarrow \text{Definición de formato} \Rightarrow \text{Prácticas de laboratorio}$$

Por razones de tiempo, que se explican más adelante, no fue posible cumplir con este camino tal cual fue programado. Por tal motivo, el camino que se siguió finalmente fue:

$$\text{Guía de secciones} \Rightarrow \text{Prototipo} \Rightarrow \left\{ \begin{array}{c} \text{Definición de formato} \\ \text{Práctica 2} \end{array} \right\} \Rightarrow \text{Evaluación} \Rightarrow \left\{ \begin{array}{c} \text{EasySim3 - ajustes} \\ \text{Práctica 3} \\ \text{Práctica 4} \end{array} \right.$$

En adelante se describen los pasos seguidos durante este proceso, comenzando con la guía de secciones. Luego se detalla el diseño del prototipo y lo que éste involucró. Finalizado el prototipo, se describen los puntos relevantes del diseño de la segunda práctica de laboratorio, así como la instancia de evaluación y sus derivaciones. Finalmente, se mencionan brevemente las dos prácticas de laboratorio restantes.

6.2.1 Guía de secciones

La guía propuesta al inicio del proyecto incluía las siguientes secciones:

1. Introducción
2. Breve desarrollo teórico
3. Bibliografía recomendada
4. Ejercicios preparatorios para la realización del laboratorio
5. Solución de los ejercicios pre-laboratorio
6. Planteo de la práctica de laboratorio
7. Ejecución de la práctica de laboratorio (Por ejemplo: cargar el Simulador)
8. Ejercicios pos laboratorio y/o evaluación del mismo
9. Solución de los problemas planteados en el punto anterior
10. Devolución al estudiante sobre su desempeño en la práctica

Durante el proceso de diseño del prototipo, se fueron notando que las secciones propuestas podían ser agrupadas de forma diferente para lograr mayor claridad. Además, algunos de los puntos numerados anteriormente no ameritaban ser una sección particular sino parte de una. Asimismo, se determinó que faltaban secciones que englobaran contenidos de integración o para profundizar los temas tratados. Luego de la evaluación se llegó a una nueva guía de secciones:

1. Introducción
2. Procedimiento
3. Fundamento teórico
4. Cuestionario previo
5. Ejecución
6. Cuestionario posterior
7. Para saber más…
8. Resumen integrador

El prototipo se estructuró mediante esta nueva guía de secciones. En primer lugar, se da una introducción al tema que se trata en el laboratorio, los objetivos y el alcance del mismo. Luego se brinda el procedimiento que deberá seguir el estudiante para realizar la práctica. El tercer punto, corresponde al material teórico necesario y/o los requerimientos previos para la realización de la práctica. En cuarto lugar, se plantea la realización de un cuestionario previo, para asegurar que el estudiante se encuentra en condiciones de realizar el laboratorio. Éste evalúa el conocimiento teórico necesario para la ejecución del mismo. Además, cumple la función de preparación para la práctica. La siguiente etapa del laboratorio es la ejecución de la práctica, en esta sección puede brindarse al estudiante un procedimiento y/o una guía de análisis de los resultados. Finalmente, se completa el laboratorio con un cuestionario posterior que brinde al estudiante una devolución de los conocimientos que se pretenden transmitir.

En forma opcional, se agregan las secciones *Para saber más* y *Resumen integrador*. La primera orientada a profundizar en las cuestiones vistas en la etapa de ejecución. La segunda busca generar un espacio en el que el docente brinde nexos a otros temas relacionados a los vistos, ya sean dentro del curso (académicos) o de la vida profesional (técnicos).

6.2.2 Diseño del prototipo

El prototipo, por ser la primera práctica de laboratorio en diseñarse, se sometió a una fuerte exigencia. La misma fue que debía ser una práctica completa, es decir, contar con todas las secciones definidas en la guía incluso las opcionales. Con este objetivo, se siguieron los sucesivos pasos:

1. Definir el tema y los objetivos de la práctica
2. Buscar material
3. Diseñar la práctica
4. Plantear los cuestionarios
5. Organizar los documentos y generar el paquete SCORM

Elección del tema y búsqueda de material

Para la elección del tema a tratar en el prototipo de laboratorio se evaluaron varios elementos. Entre ellos las capacidades disponibles en el simulador, los temas que se tratan en los laboratorios del curso de redes de datos, la importancia del control de congestión en las redes actuales, y los protocolos que se encuentran en evolución. De éstos, se decidió trabajar sobre los mecanismos de control de congestión que utiliza el protocolo TCP, en particular el algoritmo de ventanas deslizantes de TCP Reno. No obstante, se decidió buscar material sobre el manejo de ventanas que realizan otras variantes de TCP para realizar un estudio comparativo.

El siguiente paso fue conseguir el material adecuado para la preparación, tanto para el planteo de los cuestionarios, como para generar el teórico necesario para el estudiante. Se investigaron varias fuentes, principalmente con búsquedas en Internet, lo que implicó una gran variedad en el material obtenido y un proceso de selección muy riguroso. Además del material proveniente de las búsquedas, como guía fundamental, se contó con el material proveniente de las asignaturas que tocan el tema y con los libros que en éstas se utilizan.

Diseño de la práctica

El prototipo, tal como se definió, debería ser estructurado como un laboratorio completo. Es así, que en la etapa de ejecución se enfocó a la realización de una red que permitiera observar los cambios en la ventana de transmisión durante una conexión con TCP Reno. Con la simulación de esta red es posible observar los cambios que se producen por causa de TDA y *Time-Out*. Además, de las evoluciones de la ventana en las etapas de *Slow Start* y *Congestion Avoidance*.

Descripción de la red utilizada

En la Figura 23 los hosts de la izquierda ofician de emisores de tráfico y el de la derecha oficia de receptor. El host_1 inicia la conexión antes que el host_3. Se estudia el comportamiento del la ventana de transmisión del host_1 desde el inicio de la conexión y su evolución al compartir los recursos de la red.

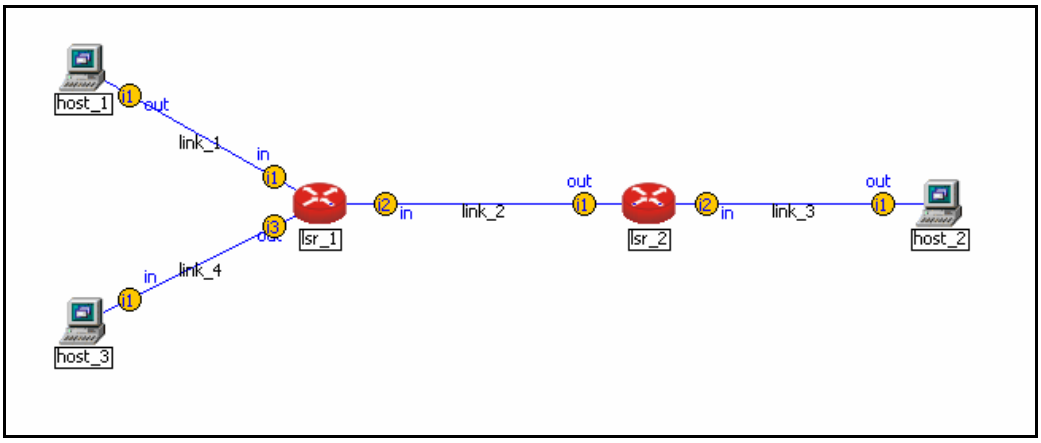


Figura 23: Laboratorio TCP – Topología

Datos de la red

Simulación: Duración 5s

Generadores

Host_1: TCP Reno 1000bps, tiempo de inicio 0s
Host_3: TCP Reno 5000bps, tiempo de inicio 1s

Enlaces

link_1: bw 1000Bps, retardo 1ms
link_2: bw 400Bps, retardo 1ms
link_3: bw 2000Bps, retardo 1ms
link_4: bw 1000Bps, retardo 1ms

Todas las colas utilizan la política de descarte *Drop Trail*. Los *buffers* tienen una capacidad de 25Kb, con la excepción del *buffer* de salida en la interfaz 2 del lsr_1 (enrutador 1) que tiene una capacidad de 7,5Kb.

En el laboratorio se incluye el código de configuración de esta red para que el estudiante simplemente la cargue en el simulador. Este se brinda en un archivo de texto plano que el estudiante debe descargar mediante un enlace en la sección Ejecución.

Por último, se decidió que con el enfoque que se deseaba dar al tema, no era suficiente con mostrar el mecanismo ventanas exclusivamente en TCP Reno. Pese a que este es el protocolo más utilizado en las redes actuales, no es el único. Por tal motivo se incluyó un anexo que compara TCP Reno con TCP Vegas. El mismo contiene el teórico necesario, dos redes a simular, y una guía de análisis de los resultados obtenidos.

Elaboración de los cuestionarios

La elaboración de los cuestionarios previo y posterior, se divide en dos etapas. La primera es la más difícil, es decir, es la que exige mayor experiencia por parte de los docentes. En ésta, se diagraman los cuestionarios en función de su cometido, o sea, se escriben las preguntas, opciones, y justificaciones para las respuestas correctas. Esta es la etapa de elaboración de la evaluación, de igual forma que los docentes deben pensar las preguntas para los parciales. Sin embargo, poseen una limitante, estos cuestionarios por el momento deben ser múltiple opción.

En el caso del prototipo se diseñaron dos cuestionarios. Ambos compuestos de cinco preguntas, con cinco opciones cada una. En el cuestionario previo, se buscó evaluar la comprensión del material teórico brindado. Para ello, se realizaron preguntas sobre el mecanismo de ventanas deslizantes y su comportamiento en ciertas situaciones. Además, se incorporaron dos preguntas sobre el funcionamiento de TCP, en lo referente a triple ack duplicado (TDA) y time-out (TO). El cuestionario posterior, se dedicó al estudio de los resultados de la simulación. Especialmente, se diseñaron las preguntas como guía de análisis, de forma que el estudiante tenga que verificar los resultados para responder el cuestionario.

La segunda etapa consistió en trasladar las preguntas antes elaboradas a los archivos de configuración para CreaQuiz. Cada cuestionario requiere dos archivos en formato html. Uno con las preguntas y sus respectivas opciones, donde cada posible respuesta debe estar asociada a un botón de opción (◌). Este archivo es el cuestionario que ve el alumno. El otro archivo, debe contener las preguntas, la justificación de la respuesta y las opciones, con el formato que se muestra en la Figura 24. A diferencia del primer archivo, este nunca será visto por los estudiantes. Este archivo es usado por CreaQuiz para determinar si las respuestas dadas son las correctas y hacer los cálculos del resultado obtenido. Además, la justificación que aquí se escriba, será la que obtendrá el estudiante como resultado.

```
num=1;
preg="¿Qué tipo de control de congestión es el balde con fichas?";
expl="La respuesta correcta es la A). El balde con fichas busca la
prevencion de la congestion controlando el tráfico";
arr[1]="prevención";
arr[2]="supervisión";
arr[3]="corrección";
arr[4]="conformación";
arr[5]="vigilancia";
ibien=1;
score += procResp(num, preg, expl, ibien, arr);
ops =5;
document.writeln("<hr>");
```

Figura 24: Edición de cuestionarios con CreaQuiz – Respuestas

Generación del objeto SCORM

Una vez superada el período de diseño y creación de cada sección del prototipo, el trabajo se centró en el armado del paquete SCORM. Para ello se utilizó RELOAD. Este programa permite crear un *IMS Content Package*, en el que se incluyen los archivos necesarios para la realización del laboratorio.

Como se explicó en la sección 5.5.2, RELOAD se divide en tres bloques. Para el amado del paquete SCORM en el primer bloque deben figurar todos los archivos a utilizar. El segundo bloque es el que se usa para editar el ***imsmanifest.xml***, archivo que define la estructura del objeto. El mismo se divide en Metadatos, Organización y Recursos. El tercer bloque es el que se encarga de asociar los recursos a la organización. La organización que se muestra en la Figura 25, corresponde a la definición de secciones realizada previamente. El nivel superior contiene el nombre de la práctica de laboratorio y los consiguientes niveles los distintos apartados que componen la práctica.

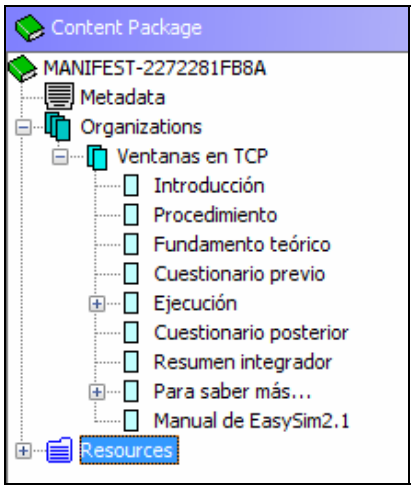


Figura 25: Edición del imsmanifest.xml – Creación objeto SCORM

La práctica que se diseñó como prototipo, una vez evaluada, fue incorporada al conjunto de productos finales como Práctica 1. La estructura del objeto SCORM final y las redes utilizadas pueden verse en el Anexo F, mientras que la práctica de laboratorio se adjunta en el CD de este proyecto bajo el nombre Ventanas.zip.

6.2.3 Laboratorio de TCP

A diferencia del prototipo antes visto, este laboratorio ya tenía estructura armada, cuestionarios previo y posterior, material teórico necesario, etc. El trabajo aquí se centro en la utilización del formato definido y en la incorporación del nuevo simulador. Los cambios en el formato no presentaron mayores problemas, ya que en definitiva era realizar una traducción de una estructura a otra, pero sin modificar significativamente los contenidos. Además, se aprovechó esta instancia para incorporar el manual rápido del simulador EasySim3.

Sustituir el simulador EasySim1 por su nueva versión EasySim3, implicó un tratamiento más delicado. En el cambio de versión del 1 al 2 se había modificado el codificado de las redes en los archivos de texto, por tal motivo la red que se incluía en el archivo SCORM base, ya no es compatible con el nuevo simulador. La re-codificación no fue sencilla. Por un lado, porque las funcionalidades cambiaron mucho entre las distintas versiones, y por otro porque cambió el formato de definición de la red. Los dos cambios de funcionalidad más relevantes fueron:

EasySim1	***EasySim3***
Ruteo estático	Ruteo dinámico
Tiempos en milisegundos	Tiempos en segundos

Que el ruteo cambie, sólo se tradujo en tener que configurar las tablas de ruteo en forma estática. Sin embargo, el cambio en la unidad de medida de tiempo provocó mayores problemas, aunque no por el cambio en sí, sino porque no existía información documentada sobre el mismo. Luego de superar estas diferencias funcionales, el trabajo se centró en las diferencias de configuración.

EasySim3, al igual que en EasySim2, trabaja con archivos de configuración en que se sigue un diseño por bloques y módulos. Primero, los parámetros relativos a la simulación, como duración y lugar en que se guardan los datos. Segundo, el posicionamiento espacial de los elementos de la red. Tercero, la definición de la

topología de red. Cuarto, una serie de bloques correspondientes a cada nodo de la red, conteniendo los parámetros por cada módulo. Son considerados módulos los generadores de tráfico, agentes de capa 4, tablas de ruteo, mecanismos de descubrimiento, y las interfaces entre otros.

EasySim1 utiliza archivos de configuración en los que se lista cada nodo a definir con nombre y coordenadas. En el caso de los *host*, al definirlos se pueden agregar los generadores de tráfico. En la definición de los *routers* se debe agregar una referencia a la tabla de ruteo que escribe explícitamente al finalizar de definir todos los elementos de la red. Luego se listan los enlaces y sus propiedades. Finalmente, se escriben las tablas de ruteo y la última línea se reserva para el tiempo de simulación.

Las diferencias encontradas entre ambos sistemas de configuración, se tradujo en que la migración de una versión a otra llevara un tiempo de análisis y prueba mayor al previsto. En particular, porque existen parámetros en EasySim3 que no estaban contemplados en las funcionalidades de EasySim1. Una vez que se tradujo toda la información disponible de un formato al otro, fue necesario levantar esta indeterminación mediante pruebas de comportamiento.

En definitiva, luego de que se obtuvo la red a utilizar, se realizó una revisión de los documentos que componían el laboratorio original. En la misma se detectaron errores menores, como opciones duplicadas en los cuestionarios, que se corrigieron. La estructura del objeto SCORM final y las redes utilizadas pueden verse en el Anexo F, mientras que la práctica de laboratorio se adjunta en el CD de este proyecto bajo el nombre LabTCP.zip

6.2.4 Evaluación

En esta instancia se expuso el trabajo realizado, hasta ese momento en el proyecto, al grupo de trabajo de Redes de Datos del IIE. En la misma se presentó el trabajo y se realizó una demostración de una práctica de laboratorio virtual prototipo. Una sección importante de los laboratorios es la etapa de simulación, en consecuencia fue necesario mostrar el simulador EasySim3 en funcionamiento.

Luego de la misma se realizó una etapa de discusión, con el objetivo de obtener realimentación del cuerpo docente sobre el formato y estructura de los laboratorios diseñados. Además, se buscó ampliar la visión sobre los temas tratados en los mismos. A continuación se brindan los principales temas tratados en la discusión.

Lluvia de ideas

Sobre las evaluaciones (cuestionarios): Buscar alternativas a la múltiple opción, posibilidad de guardar las respuestas abiertas en Moodle, limitar la realización de las evaluaciones a una única vez (esto ya se puede hacer con Moodle). Es difícil conseguir cuestionarios abiertos que sean independientes de la plataforma, ya que no se han desarrollado herramientas compatibles con SCORM en esta área.

Alcances del simulador: No es posible actuar en base a la carga útil de los paquetes, ya que ésta se representa únicamente como un valor. El simulador no maneja direcciones IP, ni nada relacionado a este protocolo (Ej. encabezado y banderas). De la capa de red, sólo cuenta con las tablas de ruteo y el protocolo OSPF. El ruteo estático se realiza mediante los nombres de los nodos y no por su dirección

IP. Por tal motivo, no es posible agrupar destinos en una misma entrada de la tabla de ruteo.

Posibles temas para laboratorios: Algoritmo de Nagle y síndrome de la ventana tonta. Manejar varios flujos con distintas prioridades de tráfico y observar el comportamiento de los *buffers* y anchos de banda asignados. Control de flujo con balde de goteo y balde con fichas, ajustando las políticas de despacho. Manejo de ventanas diferente al realizado por TCP, como *stop and wait* (requeriría implementar un nuevo protocolo de capa 4 en el simulador). Comparación de comportamiento entre UDP y TCP con pérdidas (se puso en discusión que de UDP solo se obtienen estadísticas); intentar explicar porque razón se usa UDP para ciertas aplicaciones.

Correcciones a realizar sobre EasySim3: Corregir el tamaño de paquete que no es coherente MTU + *Header*. Corregir los números de secuencia que aparecen por paquetes y deberían ser en Bytes, ya que el tamaño de paquete no es fijo.

Procesamiento de Ideas

La discusión sobre las evaluaciones y el alcance del simulador, no provocaron cambios sobre el trabajo realizado. Sin embargo, sirvió para dejar en evidencia que aún se puede mejorar mucho. En relación a los temas sugeridos para las siguientes prácticas de laboratorio, fue necesario realizar un análisis de viabilidad, complejidad, y utilidad de cada uno.

- **Algoritmo de Nagle y síndrome de la ventana tonta.** Este tema fue descartado ya que se ve en detalle en un laboratorio presencial del curso Redes de Datos.
- **Manejar varios flujos con distintas prioridades de tráfico y observar el comportamiento de los *buffers* y anchos de banda asignados.** Es una práctica que puede ser implementada sin dificultad en EasySim3, pero la visualización del fenómeno no es tan sencilla. El usuario debe tener muy buen manejo del simulador para trabajar con múltiples gráficas, separadas por prioridades de tráfico e interfaces, y con ello poder observar la utilización de los anchos de banda. Por los motivos mencionados se descartó la idea.
- **Control de flujo con balde de goteo y balde con fichas, ajustando las políticas de despacho.** El simulador maneja varias disciplinas de despacho y políticas de descarte. Manejando estas variables y los anchos de banda de los enlaces es posible simular estos algoritmos. La visualización de los mecanismos se realiza con sólo dos gráficas, en las interfaces de salida nodo que lo implemente. Además, la práctica se consideró de interés y por lo tanto fue uno de los temas elegidos.
- **Manejo de ventanas diferente al realizado por TCP, como *stop and wait*.** Requeriría implementar un nuevo protocolo de capa 4 en el simulador. En esta etapa del proyecto no era razonable seguir modificando código del simulador, por tal motivo se descartó esta idea.
- **Comparación de comportamiento entre UDP y TCP con pérdidas.** El simulador EasySim3 no genera archivos de salida para UDP. La comparación de los protocolos sólo basada en estadísticas no parece suficiente. Sería de mayor interés, si se pudiera obtener más información, como por ejemplo la distribución de las pérdidas y no sólo el valor total de paquetes perdidos. Esto es realizable,

pero nuevamente requiere que se modifique código del simulador. En conclusión, la idea se descartó en esta etapa.

Correcciones sobre EasySim3

Se analizaron las sugerencias recibidas. Teniendo en cuenta el fin pedagógico del simulador, se decidió que las sugerencias deberían ser implementadas. La primera sugerencia, referida al tamaño de los segmentos, fue la más sencilla de implementar. Simplemente se buscaron los métodos encargados de la creación de segmentos y se cambió el tamaño de estos de SEG_SIZE a HEADER + SEG_SIZE.

La segunda sugerencia apuntaba a que los números de reconocimiento se manejaran de la misma manera que en la realidad. El simulador los manejaba en función de los números de segmentos. Mientras que, en la realidad se manejan en bytes. O sea, en función del tamaño del segmento recibido. Para ello, fue necesaria la creación de variables auxiliares que lleven el control de los paquetes enviados para el manejo de las ventanas y retransmisiones. De esta forma, se dejó libre de esta tarea a las variables sn (número de secuencia) y ack (número de reconocimiento) que pasaron a medirse en bytes.

6.2.5 Laboratorio de OSPF

El tema de esta práctica de laboratorio surge por dos motivos. El primero examinando las nuevas funcionalidades de EasySim3, en particular el ruteo y la importancia del estudio dinámico de redes. El segundo motivo se relaciona directamente al desarrollo del proyecto. En la primera etapa se habían presentado problemas en el funcionamiento del protocolo OSPF, por lo que se diseñó esta práctica como forma de verificar su correcto funcionamiento.

Se planteó como objetivo estudiar una red en forma dinámica, en la cual se puedan observar la creación de las tablas de ruteo, así como su actualización ante cambios en la red. Asimismo, se quiere mostrar en funcionamiento el mecanismo de descubrimiento de OSPF y el cálculo de caminos mínimos utilizando el algoritmo de Dijkstra.

El material teórico que se utilizó provino de muchas fuentes, en particular libros como [5], [33], y [34]. Asimismo, para la sección en que explica el algoritmo de Dijkstra se utilizó material proveniente de la Universidad Carlos III de Madrid, Departamento de Ingeniería Telemática [11].

Esta práctica es netamente de visualización de fenómenos. A tales efectos se utilizó la red de la Figura 26, donde host_1 es el emisor y el host_ 2 el receptor. Se trabaja con el protocolo UDP, para evitar el hecho de que las conexiones TCP se interrumpen al ocurrir muchas pérdidas.

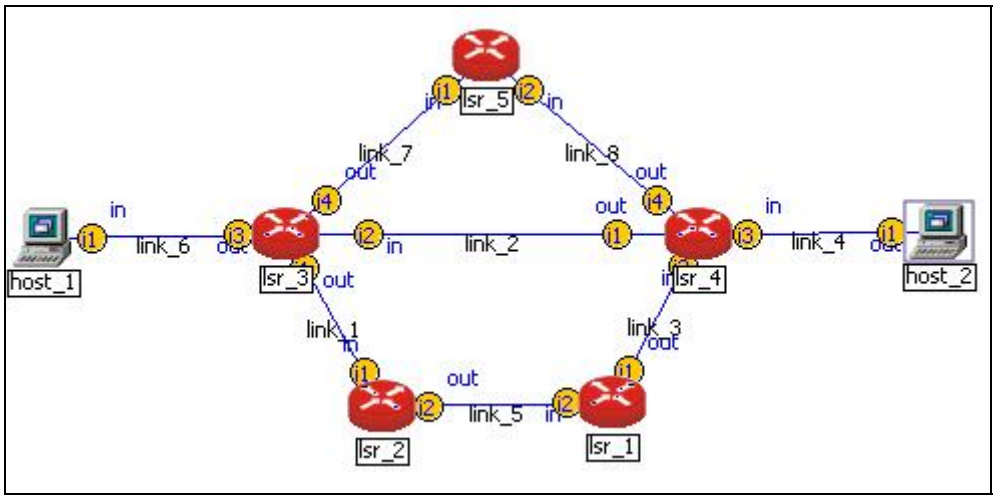


Figura 26: Laboratorio OSPF – Topología

Datos de la red

Simulación: Duración 40s

Generadores:

Host_1: UDP 1200Kbps, tiempo de inicio 0s

Enlaces

Salvo el link_2 Todos los links tienen las siguientes características: bw 1000Bps, retardo 1 ms.

Link_2: bw 1000Bps, retardo 2 seg.

OSPF

k-hello-interval 4 seg
hello-interval 2 seg
Métrica: *Minimum Hop Routing*.

Todas las colas utilizan la política de descarte *Drop Trail*. Los *buffers* tienen una capacidad de 25Kb. Todos los enrutadores actualizan las tablas cada 2 segundos, comenzando a los 0 segundos.

En el laboratorio se incluye el código de configuración de esta red para que el estudiante simplemente la cargue en el simulador. Este se brinda en un archivo de texto plano que el estudiante debe descargar mediante un enlace en la sección Ejecución.

Para llegar a los objetivos planteados, el estudiante debe realizar la simulación en modo gráfico y visualizando las tablas de ruteo en paralelo. Específicamente, la tabla de ruteo del lsr_3, ya que los cambios en la red se dan a causa de la caída del link_2 y su posterior restauración. En este sentido, la práctica tiene una sección en que se brindan las instrucciones para su realización.

De acuerdo a la definición de formato, se crearon dos cuestionarios. Ambos compuestos de cinco preguntas, con cinco opciones cada una. En el cuestionario previo, se buscó preparar al estudiante para la práctica. En él se plantean situaciones en las que interviene OSPF y se le solicita que las resuelva. En el cuestionario posterior, se busca reforzar los conceptos vistos en la práctica, mediante observaciones puntuales de hechos de la simulación. Además, se plantean tres

preguntas sobre una red más compleja, en la que corre OSPF pero con otra métrica. Se deja al alumno como opcional, implementar esta red en EasySim3 para verificar sus resultados.

Una vez creados todos los archivos, y codificados los cuestionarios para CreaQuiz, se generó el paquete SCORM correspondiente. La estructura de dicho paquete y las redes utilizadas pueden verse en el Anexo F, mientras que la práctica de laboratorio se adjunta en el CD de este proyecto bajo el nombre LabOSPF.zip.

6.2.6 Laboratorio de prevención de congestión

A diferencia de las prácticas antes descritas, esta no parte de una inquietud del grupo. El tema que en ella se estudia, fue un resultado de la presentación que se efectuó a docentes del IIE. La práctica analiza, en forma general, distintos mecanismos para acondicionamiento de tráfico como parte de una política de control de congestión. En ella se observan ventajas y desventajas de los dos algoritmos más utilizados a estos efectos: balde con fichas (*token bucket*) y balde con goteo (*leaky bucket*).

El objetivo de esta práctica de laboratorio es analizar los efectos que tienen sobre el tráfico, la aplicación de los algoritmos antes mencionados. Así como, su utilidad para la prevención de congestionamiento. No obstante, también se busca que el estudiante pueda determinar que desventajas tiene el uso de cada uno y para que aplicaciones son adecuados.

La bibliografía hallada en relación al tema es muy amplia, aunque en el nivel que se buscó trabajar, lo más útil resultó apegarse a los textos tradicionales. Es así que la mayor parte del material utilizado provino de “Redes de Computadoras” de A. Tanenbaum [5] y de “Redes de comunicaciones” de A. Bianchi [8].

Con el objetivo de limitar el estudio únicamente a los algoritmos, se diseñó la red más simple posible. En la Figura 27, se brinda la red junto al esquema de algoritmos que ésta implementa. El host_1 posee un generador que intenta inundar la red, además utiliza un balde con fichas en su interfaz de salida i1. Mientras que el lsr_1 implementa en su interfaz de salida i2 un balde con goteo. Se utiliza como red genérica un único *host* (host_2) que se limitará a responder los mensajes recibidos del host_1.

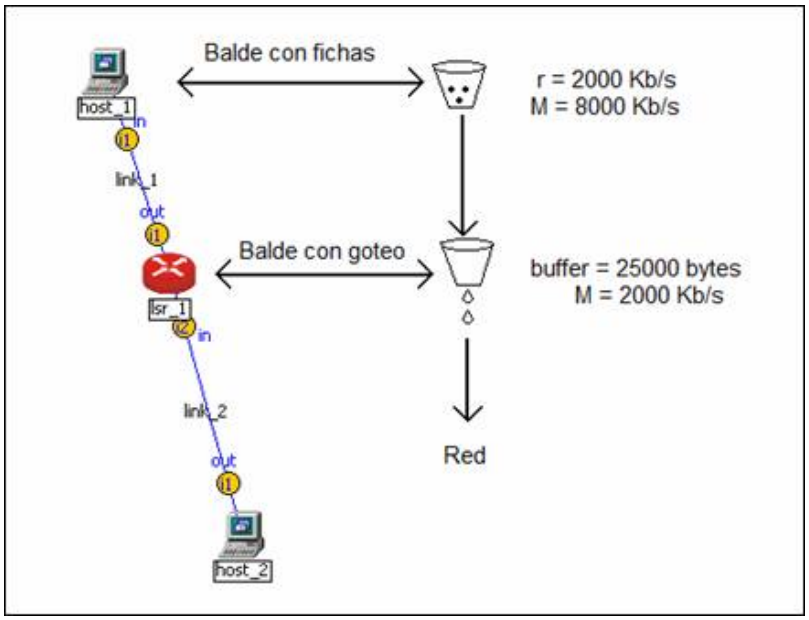


Figura 27: Laboratorio Prevención de congestión – Topología

Datos de la red

Simulación: Duración 3s
Generadores:
Host_1: TCP 250MBps, tiempo de inicio 0s
Enlaces
Link_2: bw 8000 Kbps, retardo 1 ms.
Link_2: bw 2000 Kbps, retardo 1 ms.

La realización de la práctica se centra en el estudio posterior de los resultados arrojados por las gráficas, por lo que la instancia de simulación puede realizarse en modo simple sin perjudicar el alcance de los objetivos. Una vez culminada la simulación, se le pide al estudiante comparar gráficamente el tráfico de salida de la interfaz i1 del host_1 con la i2 del host_2. (Ver Figura 28)

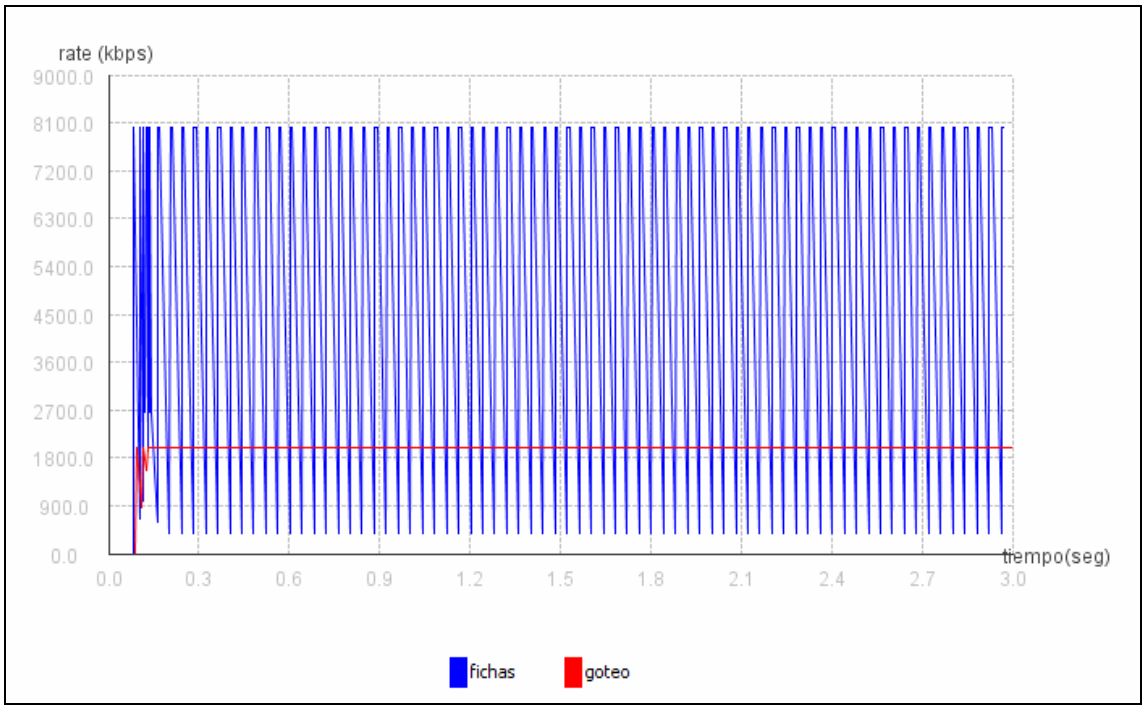


Figura 28: Algoritmos balde con fichas y balde con goteo

De acuerdo a la definición de formato, se crearon dos cuestionarios. Ambos compuestos de cinco preguntas, con cinco opciones cada una. En el cuestionario previo, se buscó preparar al estudiante para la práctica. En él se plantean una serie de preguntas sobre el teórico y pequeños cálculos. En el cuestionario posterior, se busca ayudar al estudiante a realizar un análisis de los puntos relevantes en la comparación de las gráficas y se lo guía a la obtención de resultados a partir de ellas.

Una vez creados todos los archivos, y codificados los cuestionarios para CreaQuiz, se generó el paquete SCORM correspondiente. La estructura de dicho paquete y las redes utilizadas pueden verse en el Anexo F, mientras que la práctica de laboratorio se adjunta en el CD de este proyecto bajo el nombre Prev_congs.zip

Capítulo 7: Resultados y Conclusiones

En este capítulo se brinda una descripción de los productos obtenidos como resultado de este proyecto. Se presentan además las conclusiones finales del trabajo, dividiéndose éstas en técnicas y generales. Las primeras buscan comparar los objetivos planteados con los productos finales. Mientras que las conclusiones generales apuntan a conocimientos y experiencia adquirida.

7.1 Productos Finales

Esta sección presenta los productos desarrollados en este proyecto. Se comienza por describir las propiedades del simulador gráfico de redes EasySim3 y las mejoras realizadas sobre el mismo. Luego, se detallan las prácticas de laboratorio implementadas y los productos adicionales desarrollados. Por productos adicionales debe entenderse un manual rápido para el simulador y una definición de formato para los laboratorios.

7.1.1 Simulador de redes EasySim3

EasySim3 es un simulador gráfico de redes de datos, de fácil manejo y bajo costo. En lo que se refiere a los protocolos implementados, EasySim3 provee diversas funcionalidades. A nivel de la capa de transporte (siempre considerando el modelo OSI), el simulador permite generar flujos de tráfico con cuatro protocolos distintos: UDP, TCP Reno, TCP Vegas y Fast TCP. Como protocolo de ruteo interno EasySim3 brinda la posibilidad de usar OSPF. Por mayor información sobre protocolos, funcionalidades e información general sobre EasySim3 ver el apartado 2.4.1 del presente informe.

Este proyecto generó una nueva versión de EasySim, llamada EasySim3. Ésta implicó algunas mejoras a la interfaz gráfica y, lo más destacable, la corrección de errores de funcionamiento. Además, se agregaron mecanismos como por ejemplo fin de conexión y *time-out* de envío en TCP. Exhaustivas pruebas realizadas previas a la entrega, aseguran un correcto funcionamiento del simulador en lo que tiene que ver con los protocolos TCP y OSPF. En el CD adjunto a este informe puede encontrarse el archivo ejecutable del simulador bajo el nombre EasySim3.jar además de la documentación del mismo.

7.1.2 Manual Rápido para usuarios de EasySim3

El Manual Rápido es un documento que se creó con el fin de que un usuario, que tenga conocimientos en redes, pueda comenzar a trabajar rápidamente con el simulador. La idea surgió por dos motivos. El primero se desprendió de las características del manual de EasySim2. El mismo es muy extenso, ya que brinda detalles de todas las propiedades y funcionalidades del simulador. El segundo motivo está más relacionado a los laboratorios que se diseñaron. Si se desea que se utilice la herramienta para simular las redes propuestas, la familiarización con el simulador no puede tomar mucho tiempo. De no ser así, se tornaría inviable la realización de los laboratorios, ya que el objetivo de los mismos es inculcar un concepto de redes, no el manejo del simulador.

Tomando en cuenta todo lo anterior se realizó un documento de siete páginas en las cuales se resumen las principales funcionalidades del simulador. Una propiedad

destacable del mismo es su carácter esencialmente gráfico, lo cual además de ayudar a comprender rápidamente la idea que se transmite, hace agradable su lectura. Este documento puede encontrarse en el CD adjunto a este informe.

7.1.3 Definición de Formato

Este producto es un documento que describe el estándar a seguir para la elaboración de las prácticas de laboratorio. Dicho estándar no es más que una estructura que, a criterio de los autores de este proyecto, debe tener una práctica para brindarle al estudiante la posibilidad de recrear una situación real de la mejor forma. Para su elaboración se tomó en cuenta que se estaba diseñando una práctica la cual el estudiante al momento de realizarla la hace en línea y sin el apoyo de docentes.

La finalidad de este producto es que las prácticas de laboratorios implementadas con este enfoque mantengan cierta coherencia. Esta definición no pretende ser una regla estricta a ser seguida en la implementación de una práctica, sino que servir de guía. Se considera que la existencia de un formato predefinido acorta los tiempos de elaboración. Se puede ver este documento en el Anexo D y en el CD adjunto.

7.1.4 Prácticas de Laboratorio

Se realizaron cuatro prácticas de laboratorio enfocadas en los siguientes temas relacionados a las redes de datos:

- TCP
- Mecanismo de ventanas
- OSPF
- Mecanismo de Prevención de Congestión

Todas estas prácticas son orientadas a *e-learning* y han sido diseñadas siguiendo el estándar SCORM para empaquetamiento de objetos de aprendizaje. Esto asegura que serán accesibles desde cualquier sistema de gestión de contenidos de aprendizaje que respete dicho estándar. A continuación se describe brevemente en que consiste cada una de ellas.

TCP

Esta práctica de laboratorio es la versión MAPER del primer laboratorio de la asignatura Evaluación de Performance en Redes de Telecomunicaciones. Lo anterior implica que partiendo de un laboratorio existente empaquetado en SCORM, se reestructuró llevándolo al formato definido en el documento “Definición de Formato”.

A pesar de que la práctica contaba con material teórico, cuestionarios, y ejecución, el simulador que ésta utilizaba era EasySim1. Por lo tanto, además del cambio de estructura, el trabajo se centró en la migración de la red utilizada para adecuarla al nuevo simulador. Dichos cambios conllevaron el estudio de las configuraciones de redes en ambos simuladores y sus diferencias, pero finalmente se generó la red necesaria y se concluyó la práctica.

En esta práctica de laboratorio se estudia el funcionamiento del protocolo TCP. El establecimiento y fin de conexión, así como su comportamiento en las distintas etapas de una conexión. Es un laboratorio de nivel básico.

Mecanismo de Ventanas

En esta práctica se busca que el estudiante logre comprender la importancia que posee el mecanismo de las ventanas deslizantes. Se hace especial hincapié en su contribución a controlar la congestión en las redes. En particular se estudia el comportamiento en TCP Reno. Como parte opcional se brinda una comparación del manejo de ventanas entre TCP Reno y TCP Vegas. La idea de esta sección es que el alumno compruebe que hay diversas formas de ejecutar ese mecanismo, y que cada una de ellas presenta ventajas y desventajas. Esta práctica es de nivel medio, y es recomendable realizar previamente la práctica de TCP.

OSPF

Esta práctica de laboratorio fue creada por dos motivos esenciales. El primero es porque se considera que el concepto de ruteo dinámico es crucial para poder entender el funcionamiento de las redes hoy en día. El segundo motivo es exponer el funcionamiento del protocolo OSPF, el cual como se mencionó anteriormente fue corregido en EasySim3.

En la práctica se presenta una situación en la cual durante la simulación hay un cambio en la topología. Se busca que el alumno pueda visualizar y corroborar que utilizando OSPF los paquetes buscan un camino alternativo, en caso de que exista, para llegar a destino. Asimismo, se manejan los conceptos de tabla de ruteo, paquetes de descubrimiento, cálculo del mejor camino, y algoritmo de Dijkstra.

Mecanismos de Prevención de Congestión

A diferencia de las prácticas antes descritas, esta no parte de una inquietud del grupo. El tema que en ella se estudia, fue propuesto por un grupo de docentes del IIE, como resultado de una instancia de discusión. En la misma, se presentaron las prácticas de laboratorio elaboradas y el formato de diseño para las futuras prácticas. Con esto se buscó obtener una realimentación y así plasmarla en las versiones finales.

La práctica analiza, en forma general, distintos mecanismos para acondicionamiento de tráfico como parte de una política de control de congestión. En ella se observan ventajas y desventajas de los dos algoritmos más utilizados a estos efectos: balde con fichas (*token bucket*) y balde con goteo (*leaky bucket*).

7.2 Conclusiones Técnicas

En esta sección se hará una evaluación enfocada a las metas planteadas al inicio del proyecto y a los resultados alcanzados al cabo del mismo. En ésta se recapitulan los objetivos iniciales y se analizan los resultados obtenidos. Para finalizar la sección, se dejan planteadas ideas para trabajos futuros. Cabe destacar que éstas fueron surgiendo a lo largo del proyecto y no fueron implementadas porque excedían el alcance del mismo.

7.2.1 Objetivos Planteados

En resumen, los objetivos planteados al comienzo del trabajo fueron:

- Corregir los errores de funcionamiento con que contaba el simulador gráfico de redes EasySim2. Se conocía a priori que el simulador tenía errores en la implementación del protocolo TCP. No obstante, teniendo en cuenta el uso pedagógico que se pretendió dar al mismo, se propuso

realizar una revisión en busca de errores de funcionamiento. Al cabo de la misma, se corregirían los problemas encontrados.

- Se propuso la implementación de prácticas de laboratorio con enfoque *e-learning*. Éstas tratarían temas variados en el área de las redes de datos y estarían empaquetadas bajo el estándar SCORM. Adicionalmente, se definiría un formato para los laboratorios y se elaboraría un manual rápido para el simulador.

7.2.2 Resultados Obtenidos y Evaluación

En lo referente al simulador de redes, se entrega una versión actualizada y mejorada con respecto a la que se tomó como base al inicio del proyecto. Se encuentran funcionando correctamente los protocolos OSPF y TCP (este último en sus versiones Reno y Vegas). No sólo se corrigió el funcionamiento, sino que también se aumentó el *stack* de protocolos disponibles con la inclusión del protocolo de capa de transporte Fast TCP.

Además, se corrigieron y aumentaron los mecanismos implementados. A saber, se corrigió el manejo de los números de secuencia por parte de TCP y se agregó el mecanismo de fin de conexión a dicho protocolo. A raíz de la gran cantidad de modificaciones realizadas sobre el simulador se consideró pertinente generar una nueva versión, ésta se llamó EasySim3.

Se implementaron cuatro prácticas de laboratorio, las cuales respetan el estándar SCORM y están enfocadas a *e-learning.* Dichas prácticas abarcan diversos temas relacionados a las redes de datos, como protocolos de control de tráfico, ruteo dinámico y mecanismos de ventanas deslizantes.

Considerando los objetivos planteados y a los resultados a los cuales se llegó, se puede afirmar que el proyecto ha concluido con éxito. Se han cumplido las etapas de un modo satisfactorio, logrando culminar el proyecto como un todo dentro de los márgenes de tiempo preestablecidos. Si bien se tardó más de lo planificado en la corrección del simulador, debido a que se encontraron más errores de lo esperado, el equipo de trabajo logró sobreponerse a esos contratiempos entregando todos los productos en tiempo y forma.

7.2.3 Trabajo a Futuro

Durante la etapa de trabajo con el simulador se detectaron ciertas carencias. O sea, funcionalidades que se podrán agregar en un futuro. También surgieron este tipo de sugerencias en la presentación realizada para los docentes (ver sección 6.2.4). Cabe destacar que estas no fueron implementadas porque se consideró que excedían el alcance del proyecto.

La incorporación del manejo de direcciones IP, versión cuatro y/o seis, por parte del simulador fue una de las principales inquietudes planteadas por los docentes. Esta funcionalidad permitirá una simulación más cercana a la realidad. Brindando además la posibilidad de usar el simulador para fines pedagógicos en los que se puedan tocar temas como asignación de direcciones IP y ruteo jerárquico.

Otras inquietudes surgidas fueron referentes a la implementación de otros protocolos de ruteo como RIP y BGP. La incorporación del primero permitiría realizar

comparaciones con OSPF, mientras que el segundo colaboraría a simular políticas de ruteo.

En lo que tiene que ver con la implementación de prácticas virtuales, dadas las características de los objetos de aprendizaje, muchos trabajos pueden ser realizados en esta línea. Se dejan cuatro prácticas implementadas y una definición de formato establecida. Algunos temas que pueden ser abordados fueron sugeridos en la sección 6.2.4, no obstante, existe una gran variedad de temas que pueden ser elegidos.

7.3 Conclusiones Finales

Al llegar a la instancia final del proyecto se pueden extraer conclusiones que van más allá de lo puramente técnico. Se considera que este trabajo ha aportado una gran experiencia al grupo. Especialmente en lo que tuvo que ver con la planificación de un proyecto de tal dimensión. Por primera vez los integrantes de este grupo se enfrentaron a tareas como realización de estimaciones a largo plazo y toma de decisiones frente a situaciones adversas.

Cabe mencionar que durante el transcurso del proyecto surgieron inconvenientes, no obstante se logró salir adelante y aprender nuevas técnicas de trabajo, lo cual fue de vital importancia para el éxito de las siguientes etapas del trabajo. Por citar un ejemplo, se perdieron cambios realizados por uso de distintas versiones del *workspace*. Debido a que todos los cambios realizados estaban debidamente documentados, se logró paliar esta situación rápidamente.

Otra área en la cual se adquirió experiencia fue en la del trabajo en equipo. Se cree que el grupo respondió de manera adecuada a los retos que se presentaron. Una característica fundamental, es que se logró un buen complemento entre los integrantes. Dadas las capacidades técnicas y preferencias de cada integrante, se pudieron distribuir las tareas de forma relativamente sencilla y equitativa. El grupo ya había trabajado previamente en conjunto en varias asignaturas, hecho que facilitó el trabajo en equipo.

Finalmente, pero no menos importante, fue necesario aprender en el área de planificación didáctica. Ninguno de los integrantes del grupo poseía experiencia docente como para enfrentar el diseño de las prácticas de laboratorio. Sin embargo, el desafío planteado, el apoyo de docentes y tutores, y la motivación por dejar una herramienta útil a las futuras generaciones de estudiantes, permitieron concluir el trabajo con satisfacción.

Referencias

[1] ADL – Advanced Distribute Learning – http://www-adlnet.gov/ [acceso 2008-03-25]

[2] Alves, Leonardo; Selios, Alejandro; Valentín, Juan P. *Documentación del proyecto de grado EasySim2*. Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay. Abril de 2006

[3] Alves, Leonardo; Selios, Alejandro; Valentín, Juan P. *EasySim versión2* Montevideo. Proyecto de fin de carrera año 2006 Universidad de la República Tutores: Belzarena, Pablo; González Barbone, Víctor

[4] Alves, Leonardo; Selios, Alejandro; Valentín, Juan P. *Manual de Usuario EasySim Revolution* Montevideo. Proyecto de fin de carrera año 2006 Universidad de la República Tutores: Belzarena, Pablo; González Barbone, Víctor

[5] Andrew S. Tanenbaum. *Computer Networks Fourth Edition.* Prentice Hall PTR 2002. ISBN 0130661023

[6] Belzarena, Pablo; González Barbone, Víctor. *Incorporación De Un Simulador Gráfico De Redes En Un Objeto De Aprendizaje Reutilizable*. Departamento de Telecomunicaciones. Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, Universidad de la República, Uruguay. http://www.euitt.upm.es/taee06/papers/SD/p39.pdf [Acceso 2008-04-28]

[7] Blackboard. Página oficial http://www.blackboard.com

[8] Bianchi, Aldo. *Redes de comunicaciones* – MScEE, Junio 2002 http://www.geocities.com/abianchi04/textoredes/ [Acceso 2008-04-28]

[9] Claroline Página oficial http://www.claroline.net

[10] Cisco. CCNA Network Simulator (CCNA Self-Study, 640-801). Boson Software Inc. Cisco Press, Bk&CD edition, Published July 2004, ISBN 1587201313. Descripción: http://www.bookpool.com/sm/1587201313 [acceso 2008-03-25]

[11] Dijkstra, Algoritmo de. *EncEjerciciosSolucion.pdf.* Universidad Carlos III de Madrid, Departamento de Ingeniería Telemática. http://www.it.uc3m.es/~pablo/asignaturas/rysc1/alumnos/04 [Acceso 2008-03-01]

[12] Evaluación de Performance en Redes de Telecomunicaciones. Página web del curso. Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay. http://iie.fing.edu.uy/ense/asign/perfredes/

[13] Extreme Programming – Página de divulgación http://www.extremeprogramming.org [acceso 2008-03-22]

[14] García, Mauricio; Madruga, Gastón; Paladino, Víctor. *EasySim* Montevideo. Proyecto de fin de carrera año 2004 Universidad de la República Tutor: Belzarena, Pablo

[15] García Peñalvo, Francisco José. *Estado actual de los sistemas e-learning*. Ediciones Universidad de Salamanca 2004. Universidad de Salamanca

[16] González Barbone, Víctor. *Creación del primer objeto SCORM.* Trabajo realizado en el marco del programa de Doctorado en Ingeniería Telemática de la Universidad de Vigo, bajo la orientación del Dr. Luis Anido Rifón Noviembre de 2006. No publicado

[17] González Barbone, Víctor. *E-Learning: Steering Through Term*. Trabajo realizado en el marco del programa de Doctorado en Ingeniería Telemática de la Universidad de Vigo, bajo la orientación del Dr. Luis Anido Rifón Noviembre de 2006. No publicado

[18] González Barbone, Víctor. *Una Aproximación Práctica a la Evaluación en Objetos SCORM*. Trabajo realizado en el marco del programa de Doctorado en Ingeniería Telemática de la Universidad de Vigo, bajo la orientación del Dr. Luis Anido Rifón Julio de 2006. No publicado

[19] Grupo de Traducción al castellano de RFC http://www.rfc-es.org [acceso 2008-02-11]

[20] Instituto de Ingeniería Eléctrica. Página principal de cursos. Universidad de la República, Uruguay. https://iie.fing.edu.uy/cursos [acceso 2008-04-17]

[21] LOM página oficial: http://tsc.ieee.org/wg12 [acceso 2008-04-10]

[22] Moodle. Homepage. http://moodle.org/ [acceso 2008-02-15]
Documentación: http://docs.moodle.org/en/Main_Page
SCORM: http://docs.moodle.org/en/SCORM

[23] Moodle. Página no oficial, con muy buena información sobre Moodle, uso, funcionamiento y su posición en el mercado.
http://www.eduestrategias.com/defmoodle.htm#inicio [acceso 2008-03-24]

[24] Redes de Datos. Página web del curso.
Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay.
https://iie.fing.edu.uy/ense/asign/redes/

[25] Redes Corporativas. Página web del curso.
Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay.
https://iie.fing.edu.uy/ense/asign/redcorp

[26] RELOAD Project. Reusable e-Learning Object Authoring and Delivery. Homepage. http://www.reload.ac.uk/ [acceso 2008-01-17]

[27] RFC página oficial - http://www.ietf.org – [acceso 2008-02-18]

[28] Robles, Gregorio; Ferrer, Jorge. *Programación eXtrema y Software Libre.* Universidad Rey Juan Carlos y Universidad Politécnica de Madrid – España Junio de 2007

[29] Rodríguez, Andrea; Sosa, Raquel. *Asistente Informático para la Generación de Objetos de aprendizaje.* Montevideo. Proyecto de fin de carrera. Año 2006 Universidad de la República Tutora: Motz, Regina

[30] Ruteo IP y tecnologías de transporte. Página web del curso.
Instituto de Ingeniería Eléctrica, Universidad de la República, Uruguay.
https://iie.fing.edu.uy/ense/asign/redes2/

[31] SCORM (Shareable Content Object Reference Model)
http://www.adlnet.gov/scorm/index.cfm [acceso 2008-03-23]

[32] TAEE06 – Tecnologías Aplicadas a la Enseñanza de la Electrónica – 2006
www.euitt.upm.es/taee06/ [Acceso 2008-04-28]

[33] Vito Amato – *Cisco Networking Academy Program: Guía del primer año* – Cisco System, Inc. Año 2000 – ISBN 1578702186

[34] Vito Amato – *Cisco Networking Academy Program: Guía del segundo año* – Cisco System, Inc. Año 1999 – ISBN 1587130025

[35] WebCT Página oficial http://www.webct.com

Acrónimos

ADL – Advanced Distribute Learning
API – Application Programming Interface
AS – Autonomous System
ATM – Asynchronous Transfer Mode
CMS – Content Management System
ER – Explicit Route
FEC – Forwarding Equivalence Class
FEL – Future Event List
FTN – FEC To NHLFE
GNU – GNU No es Unix
IEEE – Institute of Electrical and Electronic Engineers
IIE – Instituto de Ingeniería Eléctrica
ILM – Information Lifecycle Management
IMS – Information Management System
IP – Internet Protocol
JVM – Java Virtual Machine
LCMS – Learning Content Management System
LDP – Label Distribution Protocol
LER – Label Edge Routers
LMS – Learning Management System
LOM – Learning Object Metadata
LSP – Label Switch Path
LSR – Label Switching Router
MAPER – Material de Apoyo Pedagógico Enfocado a Redes
MIT – Massachusetts Institute of Technology
MPLS – Multiprotocol Label Switching
MTU – Maximum Transfer Unit
NHLF – Next Hop Label Forwarding
NHLFE – Next Hop Label Forwarding Entry
OSI – Open Systems Interconnections
OSPF – Open Shortest Path First
PHP – PHP Hypertext Pre-processor
RELOAD – Reusable E-Learning Object Authoring & Delivery
RFC – Request For Comments
SCORM – Sharable Content Object Reference Model
TCP – Transmission Control Protocol
TDA – Triple Duplicate Acknowledgement
TO – Time Out
UDP – User Datagram Protocol

Información adicional del Proyecto MAPER

Proyecto de fin de carrera de Ingeniería Eléctrica: Material de Apoyo Pedagógico Enfocado a Redes de datos.

Fecha de inicio: 15 de Marzo de 2007

Fecha de fin: 30 de Abril de 2008

Integrantes

- Gastón Arismendi Pereyra – gaston.arismendi.p@gmail.com – 708 45 89
- Luís A. da Rosa Fros – luis.fros@gmail.com – 708 45 89
- Rosana M. Sosa Ferreira – rosi.sosa@gmail.com – 215 36 22

Contenido de la entrega Final

- Informe final
- Repartido de Anexos
- CD conteniendo todo el proyecto
- Artículo
- Afiche