Aprender la arquitectura TCP/IP programándola

J.M. Rivadeneyra, A. González, J. Pérez, A. Mendiburu
Departamento de Arquitectura y Tecnología de Computadores
Universidad del País Vasco/Euskal Herriko Unibertsitatea
{pepe, anaisabel, txus, alex}@si.ehu.es

J. Aneiros
Facultad de Informática
Universidad de Cienfuegos "Carlos Rafael Rodríguez"
aneiros@ucfinfo.ucf.edu.cu

Resumen

En este artículo se describe el diseño de una serie de laboratorios para apoyar la docencia de redes TCP/IP. Dichos laboratorios son la base para un enfoque constructivista de esa docencia. El objetivo que se persigue con estos laboratorios es que el alumno aprenda las interioridades de la pila de protocolos TCP/IP a medida que la construye él mismo. Conforme se van analizando los problemas fundamentales de la comunicación entre computadores se van añadiendo niveles a la arquitectura de comunicaciones. Y cada vez que se añade un nivel, el alumno debe implementarlo. Todos los laboratorios están preparados en lenguaje C y basados en programación con sockets, aunque esto último queda oculto al alumno hasta el momento en que se alcanza la cumbre de la pila de protocolos.

Palabras claves: arquitectura TCP/IP, sockets, constructivismo.

Abstract

In this paper is described the design of a series of laboratories to support teaching of TCP/IP networks. These laboratories are the base for a constructive approach of teaching. The objective is the student to learn the ins and outs of the TCP/IP stack protocol while she builds it by herself. As every fundamental problem in computer communications is analysed, new levels are added to the network architecture. And whenever a level is added, the student has to implement it. All the laboratories are prepared in C, and are based on the sockets interface, although this remains hidden to the student until the top of the stack protocol is reached.

Keywords: TCPI/IP architecture, sockets, constructivism

1 Introducción

En la mayoría de las universidades del mundo, la enseñanza de asignaturas sobre redes de computadores se basa en la arquitectura usada en Internet, es decir, la arquitectura TCP/IP. Hasta no hace mucho, ese protagonismo académico lo venía ostentando el modelo de referencia OSI, que finalmente también se ha visto desplazado de este terreno por TCP/IP. Este fracaso académico de OSI es consecuencia directa de su falta de éxito en la realidad, que ha hecho que el estudiante no tenga referencias en el mundo de las redes cuando estudia los conceptos basándose en la arquitectura OSI. Con la adopción de TCP/IP ese problema queda solventado, pero aún así, otros obstáculos permanecen, dificultando la asimilación de los conceptos fundamentales relacionados con las arquitecturas de comunicaciones. Restringiéndonos a la naturaleza de las materias de estudio, dos son los principales de esos obstáculos:

1. La propia dificultad de los conceptos básicos que el estudiante debe asimilar, y la falta de un conocimiento previo de los problemas antes de proceder a su abstracción. La estructuración por niveles (o capas) del problema de la comunicación, la identificación de los problemas, conceptos como interfaz entre niveles, servicio, o protocolo, son ideas forjadas por expertos en la materia que han pasado años trabajando en comunicar computadores. Captar esos conceptos es muy difícil para un alumno que se enfrenta por primera vez al estudio de las redes de computadores, máxime cuando esos conceptos es lo primero que se le presenta al iniciarse en la materia, por ser éstos la base sobre la que se organiza todo el conocimiento posterior. Es decir, al alumno se le pide que adquiera *a priori* unos conceptos que han sido creados *a posteriori*, sobre la base de un amplio conocimiento práctico del que el alumno carece.
2. La aridez de la exposición de los protocolos de comunicación. Explicar el contenido y significado de los campos de una cabecera es una tarea poco gratificante para el docente y que no suele aportar mucho al discente. Pocas veces capta este último las razones que han llevado a los diseñadores a incluir tal o cual campo en una cabecera, lo que conlleva que, para un alumno, las características de un protocolo de comunicación sean arbitrarias. Esto dificulta enormemente la asimilación de esas características.

Para terminar de configurar el paisaje de la enseñanza superior sobre redes de computadores, hay que señalar otro aspecto que probablemente pueda generalizarse a la enseñanza superior de la informática en todos sus campos: el abuso de la clase magistral. En esta metodología docente, el papel del profesor es fundamentalmente el de hacer de transmisor de conocimientos. Un enfoque así tiene sentido cuando no existen o no están al alcance de los alumnos materiales docentes que puedan asumir ese papel transmisor. Sin embargo, ha mejorado mucho la calidad de las publicaciones disponibles acerca de redes de computadores, y hay disponible a través de Internet gran cantidad de documentación que puede sustituir con ventaja al profesor como fuente de conocimiento. Por otro lado, la ya mencionada intrínseca aridez de la exposición de buena parte de esos conocimientos, hace aún más difícil el éxito de la clase magistral.

Esto nos lleva a plantear un enfoque constructivista [1] como más apropiado para la docencia en redes de computadores. Bajo la óptica constructivista el aprendizaje no se basa en la recepción de información, sino en la construcción de la misma por parte del estudiante. Por tanto, la actividad del alumno es la base de todo aprendizaje, el papel del profesor pasa a ser primordialmente de orientador y ayudante en lugar de transmisor de información, y se debe disponer de material docente que permita poner en práctica y ensayar situaciones reales.

En este artículo vamos a describir el material de laboratorio desarrollado en el Departamento de Arquitectura de Computadores de la UPV/EHU[*] para la asignatura Redes de Computadores II, en la que se adquieren los conocimientos fundamentales relacionados con arquitecturas de comunicaciones. Ese material se ha creado como elemento necesario para poner en marcha una metodología docente basada en el enfoque constructivista esbozado en el párrafo anterior. En el siguiente apartado presentaremos someramente esa metodología. El apartado 3 es la columna vertebral del artículo: en él se describen los laboratorios, con comentarios acerca de su implementación. En el apartado 4 comentamos las experiencias habidas en el uso de este material en distintos entornos docentes. Finalmente, en el último apartado, extraemos conclusiones de todo lo tratado y dibujamos algunas tareas para mejorar en el futuro.

2 Metodología docente

El eje sobre el que se desarrolla la docencia son los laboratorios, siguiendo la dinámica inversa a la tradicional, en la que primero se imparte la teoría y luego se busca el afianzamiento de los conceptos teóricos mediante la práctica con ellos en el laboratorio. En nuestra metodología el alumno empieza por enfrentarse en el laboratorio a la resolución

[*] Y completado con aportaciones desde la Facultad de Informática de la Universidad de Cienfuegos.

de un problema real, para después extraer (construir) la teoría a partir de su experiencia. Cada tema se desarrolla siguiendo la siguiente ordenación:

- Paso 1: planteamiento de un problema y diseño de su solución (en el aula).
- Paso 2: implementación de esa solución en el laboratorio.
- Paso 3: estudio de las soluciones reales al problema (en el aula).
- Paso 4: resolución de ejercicios de pizarra (en el aula).
- Paso 5: trabajo personal del alumno, basado en las orientaciones dadas en el paso 3 y en una colección de ejercicios propuestos.

Habrá un tema por cada uno de los niveles de la arquitectura TCP/IP, de manera que en cada capítulo hay un problema fundamental a resolver, el mismo que se resuelve en cada uno de los niveles de la citada arquitectura (con la excepción del nivel de aplicación, donde los problemas son muchos y muy variados). Ese problema fundamental, de forma simplificada, es el que se plantea al alumno en el paso 1, antes de acudir al laboratorio. En este paso la labor del profesor es fundamental para lograr: a) que el alumno entienda el problema a que debe enfrentarse, y, b) que el alumno diseñe una solución.

En el paso 2, ya en el laboratorio, se debe implementar la solución diseñada. No es práctico ni didáctico implementar tantas soluciones como los alumnos propongan, por lo que el profesor debe ser capaz de conducir a los alumnos hacia una solución única que todos deberán implementar. Una vez que los alumnos han trabajado el problema programándolo, serán capaces de comparar con otras soluciones y de valorar las distintas características que presenten diferentes diseños. Es el momento de entrar en el paso 3, donde se analizan las soluciones reales, los protocolos del mundo TCP/IP.

Las sesiones correspondientes al paso 3 son equivalentes a las tradicionales sesiones magistrales donde el profesor describe los elementos de la arquitectura TCP/IP. En nuestra metodología pretendemos reducir el número de esas sesiones y cambiar su contenido. En lugar de ser clases descriptivas, donde más o menos exhaustivamente se detalla lo concerniente al nivel TCP/IP en cuestión, han de ser clases de orientación para el estudio personal que debe hacer el alumno. En ellas se le indica con precisión en qué partes de la bibliografía va a encontrar tal o cual descripción, pero no se hace la descripción en sí. Se comentan los aspectos más destacables de un protocolo, pero no se describe éste en detalle. Por ejemplo, se indica cómo IP resuelve tal o cual problema, o cuáles son los protocolos de encaminamiento usados en Internet, y dónde están descritos de forma comprensible, pero no se describe con detalle la cabecera IP, ni los mencionados protocolos de encaminamiento. Para poder actuar así es indispensable que el alumno tenga a su alcance una buena y reducida bibliografía. Ambas condiciones se dan en la bibliografía de esta asignatura, que puede reducirse a dos o tres buenos libros que casi todos, además, están en castellano.

El paso 4 no se diferencia de las tradicionales clases de ejercicios. En ellas el alumno se ejercita ya con las soluciones reales y no con el diseño implementado en el laboratorio.

Con esta metodología se pretende que el profesor dedique su tiempo no a preparar apuntes o transparencias (bastante buenos son los libros que ya hay [3][7][5]), sino a preparar laboratorios y ejercicios, que es de lo que no disponen los alumnos en la bibliografía. Y que el alumno, en lugar de estudiar los apuntes tomados en clase, que estudie directamente los libros de donde hasta ahora preparaba el profesor las clases. Para ese estudio contará ahora con la base de su experiencia personal en la resolución de los problemas, además de la orientación concisa del profesor en el uso de la bibliografía.

El material docente necesario para seguir esta metodología es el siguiente:

- Documentación y programas de laboratorio, para el paso 2.
- Guías de estudio para el paso 3. En ellas deben relacionarse muy detalladamente los conceptos a estudiar y dónde pueden encontrarse éstos en la bibliografía.
- Enunciados y soluciones de ejercicios para el paso 4. Planteamos que los alumnos tengan disponibles las soluciones de los ejercicios planteados y resueltos por el profesor en el paso 4. De esta manera se mejora el nivel de atención a esa resolución.
- Enunciados de ejercicios para trabajo personal del alumno.

En este trabajo nos vamos a centrar en los programas de laboratorio necesarios. Veremos cuáles son esos programas, su objetivo didáctico, cómo usarlos para alcanzar ese objetivo, y cómo están hechos.

3 Descripción de los laboratorios

Los niveles inferiores, aquellos que están por debajo de IP, son objeto de estudio en otra asignatura, Redes de Computadores I. Por ello no han sido incluidos de momento en este material, con lo que el conjunto de temas abarcados con esta metodología son los siguientes:

1. Introducción: en ella se hace un repaso de los aspectos fundamentales estudiados en la asignatura Redes de Computadores I, y se introducen los conceptos de aplicación distribuida, modelo cliente/servidor, arquitectura de comunicaciones, nivel, protocolo de aplicación e interfaz.
2. Interconexión de redes: se estudia la problemática de la interconexión de redes heterogéneas, y el protocolo IP y su *séquito* (ARP, ICMP, RIP, OSPF, CIDR, NAT, ...).
3. Control de la interred: se estudian los problemas y protocolos del nivel de transporte.
4. Aplicaciones distribuidas: se estudia el desarrollo de aplicaciones distribuidas basado en la interfaz de sockets, y las aplicaciones más importantes de Internet (DNS, correo, web ...).

Se ha desarrollado un laboratorio por cada uno de esos temas, con excepción del último, al que se dedican dos laboratorios: uno como toma de contacto con la interfaz de sockets, y otro para la implementación de un protocolo de aplicación real de Internet.

Todos los laboratorios se desarrollan en lenguaje C [2].

3.1 Laboratorio 1: Arquitectura de tres niveles

Es el laboratorio asociado al tema de introducción. El alumno deberá escribir el cliente de una aplicación para la distribución de ficheros a través de una red Ethernet (la del laboratorio). Se usará una arquitectura de comunicaciones de tres niveles: físico, ethernet, y aplicación (ver figura 1). Por tanto, lo que tiene que hacer el alumno es implementar parte del nivel de aplicación. Esa arquitectura de tres niveles ha sido diseñada previamente en el aula, en el paso 1 de la metodología, partiendo de los conocimientos que el alumno tiene sobre redes de computadores. Esos conocimientos se reducen a los aspectos relacionados con los niveles que están por debajo de IP en la arquitectura TCP/IP, a los que se añaden en la introducción unas nociones mínimas sobre aplicaciones distribuidas y arquitecturas de comunicaciones.

Obsérvese que en el contexto que se plantea al alumno (intercambio de ficheros en una Ethernet), añadiendo una cierta simplificación (ignorando pérdidas en la red y considerando que un solo proceso trabaja con la red en las máquinas cliente y servidor), son innecesarios los niveles IP y de transporte, que son, todavía, desconocidos para el alumno.

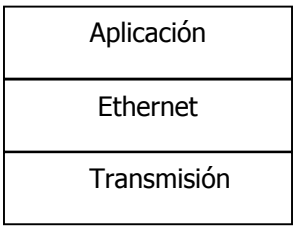

Figura 1: niveles de la arquitectura programada en el primer laboratorio.

La parte servidora de la aplicación, que se ejecuta en una máquina controlada por el profesor, gestiona una serie de ficheros que los clientes pueden descargarse. La interfaz de usuario que el cliente presenta al usuario es muy sencilla, y ya se le proporciona programada al alumno. Él tiene que hacer la otra parte del cliente: la interacción con el servidor, de acuerdo con el protocolo de aplicación definido.

Objetivos docentes del laboratorio

Son dos:

1. Afianzamiento de los conceptos de arquitectura de comunicaciones, nivel, interfaz, protocolo, aplicación, cliente, servidor.
2. Familiarización con el entorno de trabajo de los siguientes laboratorios.

Protocolo de aplicación

En el aula (paso 1) se ha definido un sencillo protocolo de aplicación. Ese protocolo de aplicación es el que deben programar los alumnos sobre la arquitectura de tres niveles diseñada. En la tabla I tenemos los comandos y

respuestas definidos en ese protocolo. La parte servidora, a cuya implementación los alumnos no tienen acceso, está programada para responder a esos comandos.

Este mismo protocolo va a ser usado en todos los laboratorios, exceptuando el último.

comando o respuesta	Parámetros/datos	Semántica
USER	Nombre de usuario (string)	Solicitud de inicio de sesión
LIST	Nada	Solicitud de la lista de ficheros disponibles
FICH	Nombre de fichero (string)	Solicitud de envío de fichero. Primer paso.
SEND	Tamaño del fichero (string)	Solicitud de envío de fichero. Segundo paso.
TERM	Nada	Cierre de sesión
OKEY	Puede llevar un parámetro indicando un tamaño de fichero (string), una serie de datos (contenido de un fichero), o no llevar nada.	Respuesta afirmativa a un comando
FINN	Puede ir acompañada de un string explicativo de por qué se ha producido la negativa.	Respuesta negativa a un comando.

Tabla I: Comandos y respuestas del protocolo de aplicación.

Interfaz de acceso a Ethernet

Todo lo que el cliente (nivel de aplicación) envíe o reciba tendrá que hacerlo a través de Ethernet. Para acceder al nivel Ethernet de la arquitectura de comunicaciones, se definen dos funciones C:

enviar_eth --> para enviar datos por la red ethernet.

recibir_eth --> para recibir datos de la red ethernet.

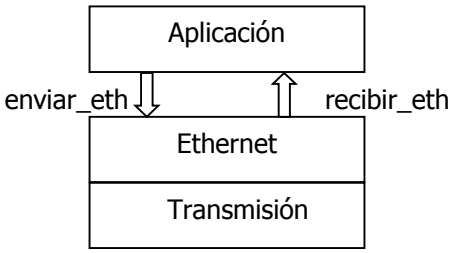

Figura 2: Interfaz de acceso al nivel Ethernet.

Estas dos funciones componen el driver de la tarjeta Ethernet. En la implementación de la arquitectura experimental definida, las aplicaciones acceden a los recursos de red (a la tarjeta Ethernet) directamente, sin la intermediación del sistema operativo. Esto tiene un objetivo didáctico: que el alumno controle al 100% todo lo relacionado con la comunicación entre ordenadores a través de la red.

La programación del driver Ethernet escapa a los objetivos de este laboratorio (concierne a aspectos estudiados en otra asignatura), por lo que se entrega ya hecha al alumno. Los fuentes en C de esos drivers se entregan también al alumno, con el único objeto de que los consulte para hacerse una idea completa de lo que ocurre en los niveles inferiores al de aplicación, y, sobre todo, para que le sirva de referencia para lo que él va a tener que programar en los dos siguientes laboratorios.

La tramoya del laboratorio

El nivel Ethernet sobre cuyos servicios se monta el nivel de aplicación en nuestra arquitectura experimental no es más que un decorado, o, dicho más técnicamente, una emulación. Las aplicaciones no van a acceder directamente a la tarjeta Ethernet (de hecho, no podrían hacerlo por falta de privilegios de ejecución), sino que van a hacerlo en la forma estándar, a través de la interfaz de sockets, atravesando toda la pila TCP/IP. Es decir, la arquitectura experimental de tres niveles está montada sobre TCP/IP como una aplicación más, a modo de arquitectura virtual. Esto queda oculto a los alumnos, que sólo 'ven' los niveles de aplicación (porque lo programan ellos) y Ethernet (que se les suministra). Si el alumno examina el código del nivel Ethernet (el supuesto driver), encontrará en él una serie de llamadas al nivel físico para emitir y recibir, a cuyo código no tiene acceso. Lo que hacen estas llamadas no es más que escribir o leer en un socket a través del cual se ha abierto una conexión con el servidor de la aplicación. Antes de escribir, se hacen algunas comprobaciones para verificar que el alumno ha hecho bien su trabajo al

construir las cabeceras, para despojar después a la información de nivel de aplicación de esas cabeceras. Al recibir, hay que construir unas cabeceras ficticias que coincidan con lo que el alumno espera encontrar en ellas.

Previo a poder enviar o recibir se debe abrir una conexión con el servidor, algo que no tiene cabida en la arquitectura de tres niveles. Para salvar esta situación se recurre al artificio didáctico de tener que configurar la red para trabajar con nuestra arquitectura experimental antes de empezar a trabajar con ella. Para ello se suministra al alumno una función *configurar_eth()* que debe ser ejecutada por su cliente antes de empezar a trabajar con la red. Lo que realmente hace esta función es crear un socket y abrir sobre él una conexión con el servidor. A partir de ese momento, las llamadas a alguna de las funciones de la interfaz Ethernet son mapeadas a llamadas de lectura y escritura sobre ese socket.

La implementación de la llamada *configurar_eth()*, así como la de las funciones de acceso al nivel físico, se suministran al alumno en un fichero objeto, como si fueran la implementación del nivel físico. El alumno deberá compilar su cliente con ese fichero y con el fichero fuente del nivel Ethernet que se le ha suministrado.

Evidentemente, la arquitectura virtual sólo está activa en la parte del cliente. El servidor, cuyo código fuente no conoce el alumno, trabaja directamente sobre la interfaz de sockets, recibiendo comandos de aplicación y enviando respuestas. Ni siquiera tiene contacto con las cabeceras Ethernet introducidas en la parte cliente, porque *la tramoya* se encarga de eliminarlas antes de enviar la información por el socket, de manera que al servidor sólo llega información de nivel de aplicación.

3.2 Laboratorio 2: Arquitectura de cuatro niveles

Este laboratorio se asocia al tema de interconexión de redes. El problema que se plantea a los alumnos en el aula (paso 1 de la metodología docente) es el de comunicar dos ordenadores situados en redes diferentes. Se recurre a la misma aplicación del primer laboratorio, considerando ahora que servidor y clientes se encuentran en redes distintas e interconectadas entre sí por un conglomerado heterogéneo de redes. La solución a este problema será, cómo no, añadir un nivel para interconexión de redes a la arquitectura anterior, con lo que ahora tendremos la arquitectura experimental de cuatro niveles representada en la figura 3a.

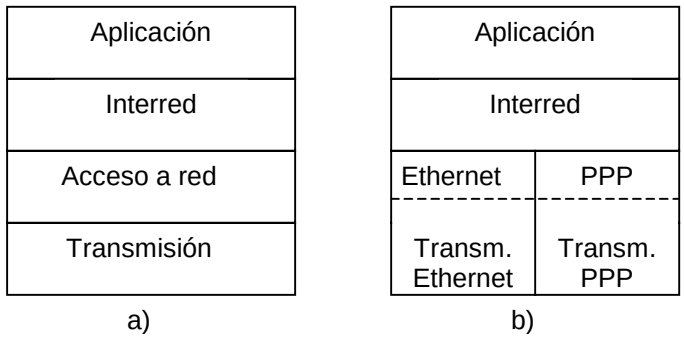

Figura 3: Arquitectura experimental de 4 niveles. a) General b) Máquina del cliente.

El alumno deberá escribir en C la implementación del nivel de interred en las máquinas donde se ejecuta el cliente. En estas máquinas se dispondrá de dos accesos a red: uno mediante Ethernet, similar al del laboratorio 1, y otro mediante una conexión PPP por línea telefónica a un proveedor de red de datos por conmutación de paquetes. Esa estructura de red es la que refleja la figura 4.

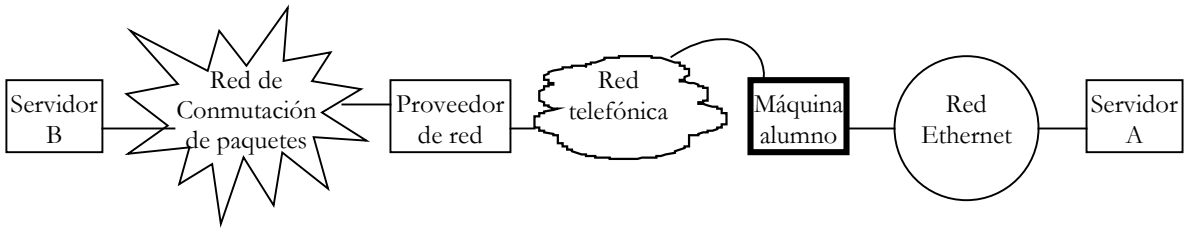

Figura 4: Estructura de red emulada en el laboratorio 4.

Se proporciona al alumno un cliente de la aplicación programada en el primer laboratorio, pero adaptado a la nueva arquitectura. Es decir, el cliente, en lugar de usar los servicios del nivel Ethernet (laboratorio 1), usará los servicios del nivel de interred de la nueva arquitectura, cuyo acceso deberá programar el alumno.

En este caso, el cliente del alumno puede interaccionar con dos servidores diferentes: uno igual que en el laboratorio 1, ubicado en nuestra red local Ethernet (el servidor A en la figura 4), y otro al que se accede a través del proveedor de red de área amplia (el servidor B de la figura).

Evidentemente, toda esa estructura de redes interconectadas entre sí es, de nuevo, un decorado didáctico. Sin embargo, el alumno no lo percibe así: para él, en el laboratorio hay físicamente una red Ethernet a la que está conectado su equipo, que también está equipado con un modem conectado a una centralita telefónica local. Lo que ocurre en realidad es que en la máquina servidora A pondremos en marcha dos servidores exactamente iguales, cada uno de ellos trabajando en un puerto. Cuando el alumno *configure* su arquitectura de 4 niveles, lo que se hará será abrir dos conexiones, cada una por un socket distinto, con cada uno de los dos servidores.

Objetivos docentes del laboratorio

Se pretende que el alumno:

1. Afiance los conceptos introducidos al presentar el problema de la interconexión de redes y al diseñar un nivel de interconexión: dirección (física y de red, estática y dinámica), encaminamiento, encaminador, interfaz física, conexión PPP, protocolo de red, servicio.
2. Repase los conceptos de arquitectura de comunicaciones, nivel, interfaz entre niveles, protocolo, aplicación, cliente, servidor.

El nivel interred

La labor del alumno va a ser programar este nivel. Esto supone haber definido antes un protocolo para interconexión de redes, lo que se hace en el paso 1 de la metodología docente. Al protocolo diseñado se le llama PI (Protocolo Interred), y no será más que una versión simplificada de IP, con una cabecera que prácticamente se reducirá a una dirección origen y una dirección destino. El protocolo define un esquema de direccionamiento basado en máscaras similar al de IP.

Fuera del protocolo PI, pero como parte del nivel de interred, hay que definir también el encaminamiento de datagramas. Acometer este problema en toda su amplitud sería excesivo y desbordaría los objetivos del laboratorio (y al propio alumno), por lo que únicamente se hace una aproximación al mismo desde la perspectiva de una máquina de usuario, y no de un encaminador. Hacerlo así elimina toda la problemática de la construcción dinámica y mantenimiento de las tablas de encaminamiento. Por tanto, lo que tendrá que programar el alumno será únicamente la construcción de una tabla de encaminamiento estática, basada en el esquema de la figura 4, y las posteriores consultas a esa tabla cuando haya que enviar un datagrama. A pesar de esta simplificación, la tabla que el alumno deberá construir y gestionar no es la típica tabla de una máquina de usuario conectada a una red, pues la máquina del alumno estará (aparentemente) conectada a dos redes distintas, Ethernet y acceso PPP, lo que se parece más a una situación de encaminador periférico que de máquina de usuario.

También como parte del nivel interred, el alumno deberá programar un sistema de mapeo entre direcciones interred y direcciones físicas. Dado que en PPP no se usan las direcciones físicas (aunque podría hacerse), esta labor consistirá en construir el equivalente a una tabla ARP, para hacer la traducción de direcciones PI a direcciones Ethernet.

La construcción de las tablas de interfaces, encaminamiento y traducción de direcciones se proporciona programada al alumno con objeto de agilizar el desarrollo del laboratorio. Sin embargo, con objeto de que el alumno no se desentienda de esa parte del problema, y llegue realmente a asimilar para qué es cada una de esas tablas y cómo se construye, cada vez que arranca su cliente se ejecuta, de nuevo, una función de configuración. Esta función guiará al alumno en la construcción manual de las citadas tablas.

Interfaces de interred y de acceso a red

El cliente que se proporciona ya programado al alumno hará uso de un par de funciones, *enviar_pi() y recibir_pi()*, para enviar y recibir algo de la interred respectivamente. Esas funciones componen por lo tanto la interfaz del servicio interred al nivel de aplicación, y son lo que el alumno deberá programar. Para implementar ese servicio interred, el alumno tendrá que hacer uso a su vez del servicio de acceso a red, o, mejor dicho para este caso, de los servicios de acceso a red, puesto que las máquinas de usuario están (aparentemente) conectadas a dos redes. En la figura 5 aparecen las funciones que componen la interfaz de que el alumno dispone para acceder a los servicios de red. En esa misma figura también tenemos la interfaz que el servicio interred ofrece al nivel de aplicación.

A diferencia de las funciones para envío, que hay una por cada interfaz físico, para recepción se dispone de una única función (*recibir_int()*). Esto ha de ser así porque no podemos saber por qué interfaz físico va a llegar un datagrama; al transmitir, sin embargo, el nivel PI decide por qué interfaz ha de transmitirse el datagrama.

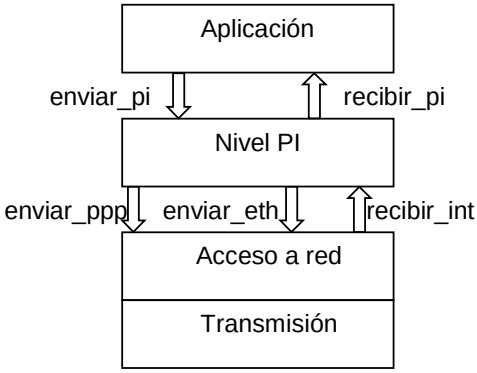

Figura 5: Interfaces de interred y de acceso a red en la arquitectura experimental de 4 niveles.

La tramoya

La implementación de las funciones de interfaz de acceso a red componen la tramoya de este segundo laboratorio. Es en ellas donde se escribe en los correspondiente sockets, o donde el cliente se bloquea a la espera de recibir algo por alguno de los sockets en los que tiene abiertas sus dos conexiones con los servidores.

Al igual que en el laboratorio anterior, para evitar que las cosas funcionen cuando no deberían, se hace también en estas funciones un cierto "control de calidad" de lo programado por el alumno. Un ejemplo de ello es la comprobación de la dirección física de destino que se hace en la función *enviar_eth()*. Si esa dirección no se corresponde con la que hemos asignado al servidor A en nuestra hipotética red, la trama ethernet que se envíe no debe ser recogida por nadie en la red, o quien la recoja la tirará a la basura sin más. Eso se simula no enviando nada realmente por aquí, lo que degenerará en que el cliente de la aplicación se bloquee posteriormente esperando una respuesta del servidor que no va a llegar, dado que el servidor no ha recibido nada.

En cuanto al servidor, hay que señalar que se usa el mismo programa que en el primer laboratorio, puesto que sólo tiene que trabajar a nivel de aplicación sobre sockets, y eso no ha cambiado de un laboratorio a otro. Únicamente se cambia el puerto por el que recibe conexiones, para permitir el trabajo simultáneo de alumnos sobre el primer y segundo laboratorio.

3.3 Laboratorio 3: Arquitectura de cinco niveles

Es el laboratorio correspondiente al tema de nivel de transporte. El problema de partida es el mismo del laboratorio 2, comunicar dos ordenadores situados en redes diferentes, pero ahora se hace teniendo en cuenta que en el trayecto interredes se puede perder información, y, además, consideraremos la posibilidad de que en una máquina haya en marcha más de una entidad de aplicación. La solución será añadir un nivel de transporte, que llamaremos nivel de control de la interred (NCI), a la arquitectura experimental, que pasará a tener cinco niveles (ver figura 6).

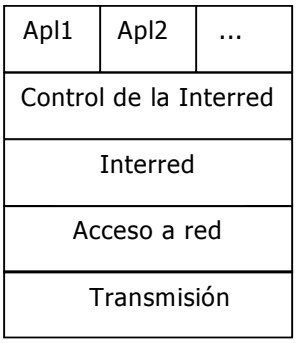

Figura 6: Arquitectura experimental de cinco niveles.

De nuevo se ofrecerá al alumno la posibilidad de interaccionar con dos servidores distintos, A y B (figura 4). Sin embargo, en este laboratorio, el hecho de que su máquina esté conectada a dos redes distintas, y tenga por tanto dos interfaces de red, resulta irrelevante para el trabajo del alumno. Toda la cuestión relativa a ese hecho es tratada a nivel interred, y por tanto el alumno se puede desentender de ello a nivel NCI.

El alumno deberá escribir en C la implementación del nivel de control de la interred. Se le proporciona un cliente de la aplicación programada en el primer y segundo laboratorio, pero adaptado a la nueva arquitectura. Es decir, el cliente, en lugar de usar los servicios del nivel Ethernet (laboratorio 1) o Interred (laboratorio 2), usará los servicios

del NCI de la nueva arquitectura. La implementación y acceso a esos servicios es lo que deberá programar el alumno.

Objetivos docentes del laboratorio

1. Afianzamiento de los conceptos de identificación de aplicación, pérdidas en los encaminadores, control de errores y de flujo, comunicación orientada a conexión, apertura activa de una conexión, apertura pasiva de una conexión, comunicación extremo a extremo.
2. Repaso de los conceptos de arquitectura de comunicaciones, nivel, interfaz entre niveles, protocolo, servicio, entidades.

El nivel de control de la interred

La referencia para el diseño de este nivel es el servicio que da TCP, es decir, un servicio fiable orientado a conexión. No obstante, con objeto de reducir la complejidad del laboratorio, todos los aspectos de control de congestiones son ignorados.

El protocolo de control de la interred, aún siendo una versión de TCP muy simplificada, es bastante más complejo que los protocolos programados en los dos laboratorios anteriores. El mero hecho de tener que abrir y cerrar conexiones ya complica las cosas, más aún si esas aperturas y cierres han de ser fiables. Pero además se introducen aquí los conceptos de apertura activa y apertura pasiva. Dado que el alumno sólo va a tener que programar la parte cliente, únicamente implementará la apertura de conexión activa, algo que simplifica enormemente su trabajo (y la tramoya).

Al tener que trabajar con distintas entidades de aplicación, surge la necesidad de identificarlas, y con ello se introduce el concepto de puerto. Para mantener clara la distinción entre nuestra arquitectura experimental y la arquitectura TCP/IP real que en el paso 3 de la metodología el alumno va a estudiar, no se referencia a los puertos como tales, sino que se usa el término *punto de conexión* para referirse a ellos.

Interfaces de control de interred y de interred

El alumno debe programar las cuatro funciones que componen la interfaz de control de la interred (ver figura 7). Estas funciones son las que usa el cliente de aplicación que se le entrega al alumno para ser compilado con el resto de la arquitectura. Para ello se apoyará en los servicios ofertados por el nivel interred a través de la correspondiente interfaz, que también puede consultarse en la figura 7.

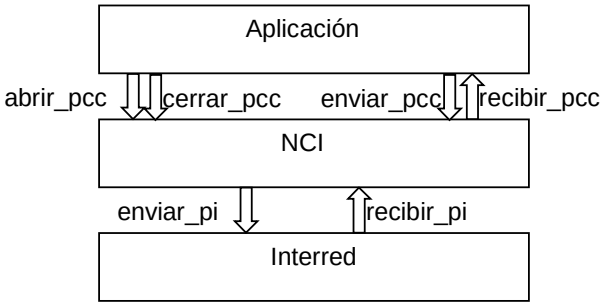

Figura 7: Interfaces de control de la interred y de interred.

Las funciones de la interfaz de interred son las que el alumno hubo de programar en el laboratorio anterior, pero para este laboratorio se le entregan versiones ya compiladas de las mismas. No obstante, en la función *recibir_pi()* se da una diferencia con respecto al laboratorio 2. Su resultado puede adoptar 3 valores: además de correcto, y error de ejecución (no de transmisión), puede devolver como resultado un agotamiento del temporizador. Este hecho supone una irregularidad, ya que es el propio nivel NCI el que debe ocuparse de gestionar sus temporizadores, y no el nivel de interred como aquí aparece. Sin embargo, en esta versión del laboratorio se ha hecho así para restar complejidad al trabajo del alumno y centrarlo en las cuestiones cruciales del diseño de un nivel de transporte. La gestión de temporizadores, siendo un problema importante, no es conceptualmente de los fundamentales en un nivel de transporte.

Tramoya

Al igual que sucedía en los laboratorios anteriores, en éste la realidad se oculta en la implementación de las funciones de acceso al nivel inmediatamente inferior al implementado por el alumno, en este caso las dos funciones de acceso al nivel interred que aparecen en la figura 7. En ellas se sigue el mismo patrón de los laboratorios anteriores: mapear los envíos y recepciones sobre las correspondientes conexiones abiertas en sockets estándar, y

hacer una serie de controles para garantizar que el alumno está haciendo lo que debe en "su" nivel. Esta segunda función se complica bastante en este laboratorio, puesto que no se va a limitar a comprobar que ciertas cabeceras están bien construidas. La gran dificultad estribará en que estas funciones tienen que emular el comportamiento de una entidad de transporte remota, controlando números de secuencia, errores, etc.

Además, y de nuevo para controlar el trabajo del alumno, periódicamente estas funciones van a introducir errores: envíos que no son realmente hechos, reconocimientos que no se entregan, timeouts falsos, etc. Cada vez que se inyecta un error de estos, se hace un seguimiento a la reacción del programa creado por el alumno. Por ejemplo, si tras un timeout el alumno no retransmite el envío afectado, en la siguiente recepción no encontrará el número de ACK esperado.

3.4 Laboratorio 4: La interfaz de sockets

Este es el primer laboratorio asociado al tema dedicado a las aplicaciones distribuidas en Internet. A lo largo de los tres laboratorios anteriores el alumno ha ido implementando la pila de protocolos de Internet, en versión didáctica simplificada. Ahora utilizará la versión real de esos protocolos, mediante la interfaz de sockets, despojándonos de todo decorado.

El alumno deberá programar en C la aplicación definida en el primer laboratorio usando la arquitectura TCP/IP. Ello supone programar tanto el cliente como el servidor. No es objetivo del laboratorio incidir en los conceptos propios de los protocolos de aplicación (ese era objetivo del primer laboratorio), por lo que no es necesario volver a programar el protocolo de la aplicación. Por ello se proporcionan al alumno sendos ficheros cliente y servidor donde ya está programado lo relativo al protocolo de aplicación, quedando pendiente de programar lo relativo al acceso a los servicios de red.

Objetivos docentes del laboratorio

1. Aprender a usar la interfaz de sockets para el desarrollo de aplicaciones distribuidas en entorno TCP/IP.
2. Aprender la estructura de clientes y servidores típicos.
3. Afianzamiento de los conceptos de puerto, comunicación orientada a conexión y no orientada a conexión, apertura activa de una conexión, apertura pasiva de una conexión.
4. Afianzamiento de la distinción entre TCP y UDP.

3.5 Laboratorio 5: Cliente SMTP

En este segundo laboratorio asociado al tema de aplicaciones distribuidas, el alumno deberá programar una aplicación de correo electrónico (parte cliente) usando la arquitectura TCP/IP, mediante la interfaz de sockets. En una sesión previa al laboratorio se describe el protocolo SMTP. No se programarán todos los comandos SMTP, sino únicamente los fundamentales, de manera similar a como se hace en [4]. Como servidor de correo se usa SendMail.

Objetivos docentes del laboratorio

1. Conocer cómo son los protocolos de aplicación en Internet, programando el caso concreto de SMTP.
2. Repaso del desarrollo de aplicaciones distribuidas basado en sockets.

4 Experiencias

4.1 Infraestructura necesaria para los laboratorios

Se necesita bastante poco para poner en marcha estos laboratorios. Con que cada alumno disponga de una máquina con un compilador de C y un sistema operativo que incluya una interfaz de sockets es suficiente. Ni siquiera es necesario, aunque sí recomendable, disponer de una red: en una misma máquina se pueden poner en marcha servidores y clientes. De hecho, a los alumnos se les ha proporcionado una versión preparada para trabajar en casa sobre un PC aislado.

Sin embargo, desde el punto de vista didáctico, es mejor que el servidor se ejecute en una máquina distinta a la del alumno, y que ambas estén en la misma red Ethernet. Si además se dispone, como así ha sido en una de las experiencias habidas, de conexión telefónica vía modem en las máquinas de usuario, la situación es ideal. Con una configuración así, el alumno llega a creer que realmente está trabajando con diferentes servidores a través de su arquitectura de comunicaciones, lo que dota a su trabajo de un realismo que incide directamente en su motivación.

4.2 Identificación de problemas

La metodología expuesta en las secciones anteriores ha sido utilizada en dos universidades durante el curso académico 01/02: en la Universidad del País Vasco (unos 120 alumnos) y en la Universidad de Cienfuegos (12 alumnos). En ambos casos los grupos de alumnos en el laboratorio han sido de entre 10 y 20 alumnos. Los principales problemas afrontados en estas experiencias han sido dos:

- La escasa preparación del alumnado para trabajar en lenguaje C. Pese a las muchas facilidades que se han dado en este aspecto, ha habido un número importante de alumnos para los que el uso de este lenguaje ha supuesto un hándicap importante. La mucha mayor motivación de los alumnos cubanos (que nunca habían tenido contacto con este lenguaje) les ha permitido superar el problema mucho mejor que sus colegas vascos.
- La sincronización entre los distintos pasos de la metodología es un aspecto de difícil solución. Normalmente, en un entorno académico regulado, los horarios y calendarios son bastante cerrados, de manera que un grupo de alumnos tiene acceso a los laboratorios en unos días y horas determinadas, que no siempre coinciden con el momento didáctico en que tienen que pasar a implementar.

El nivel de satisfacción alcanzado por los alumnos ha sido medido mediante encuestas anónimas en la UPV y recogiendo personalmente sus impresiones en el caso de la Universidad de Cienfuegos. En el primer caso, aunque la valoración general es positiva, no se alcanza la unanimidad que se da en el segundo al mostrar la satisfacción del alumnado.

5 Conclusiones e ideas para el futuro

Los laboratorios presentados en este trabajo son una herramienta muy útil para el aprendizaje de los conceptos relacionados con la arquitectura TCP/IP, basándonos en las ideas del constructivismo. Su puesta en marcha y gestión es bastante fácil: la infraestructura física que se necesita es mínima, y el software enormemente sencillo. Su bondad didáctica radica en que el alumno aprende los conceptos fundamentales relacionados con la arquitectura TCP/IP a medida que implementa una versión simplificada de esta arquitectura.

El uso de estos laboratorios ha revelado áreas que pueden mejorarse y desarrollos que completan los objetivos docentes buscados. De momento, para un futuro cercano, hemos identificado las siguientes tareas:

- Mejora de la implementación actual. Diversos aspectos de la tramoya en los tres primeros laboratorios pueden ser mejorados. Concretamente, se pueden incluir más controles del trabajo que hace el alumno, y la gestión de los temporizadores y los puntos de conexión en el laboratorio 3 es manifiestamente mejorable.
- Dejar a disposición pública el código fuente y documentación relacionada con los laboratorios, para permitir a otros docentes acceder a este material y participar en su mejora.
- Crear una versión en Java de los mismos laboratorios.
- Preparar laboratorios similares para los niveles inferiores de la arquitectura. Siguiendo la misma filosofía de estos laboratorios, se trabajaría sobre diferentes arquitecturas de 2 niveles: aplicación sobre ATM, aplicación sobre Frame Relay, aplicación sobre enlace telefónico, aplicación sobre Ethernet, etc. En ellos el alumno implementaría el nivel inferior.
- Preparar laboratorios similares al 5 para otras aplicaciones estratégicas de Internet: DNS, OSPF...
- Estudiar la posibilidad de incorporar a la metodología el análisis de implementaciones reales de la pila TCP/IP, de modo similar a como se hace en [6] pero basándonos en implementaciones sencillas de Linux.

Al margen de desarrollos técnicos, se está estudiando hacer un seguimiento de los resultados académicos de los alumnos en la asignatura durante el curso 02/03, para relacionarlos con su participación en los laboratorios. El objetivo de este estudio es hacer una medición del impacto académico de la herramienta. La hipótesis de partida es que los alumnos que participan en los laboratorios propuestos obtienen mejores resultados académicos en la asignatura, y ello independientemente de que lleguen a culminar o no los programas propuestos en cada laboratorio.

Referencias

[1] Barberà Gregori, Elena. *El constructivismo en la práctica*. Barcelona: Graó, 2000.

[2] Kernighan, B.W. Ritchie, D.M. *The C programming language.* Prentice Hall Software Series, 1978.

[3] Kurose, J.F. Ross, K.W. *Computer Networking*. Addison Wesley, 2001.

[4] López, A. Novo, A. *Protocolos de Internet, cap. 22*. Ra-Ma, 1999.

[5] Stallings, W. *Comunicaciones y Redes de Computadores, 6ª ed*. Pearson Educación, 2000.

[6] Stevens, W.R. Wright, G.R. *TCP/IP Illustrated, Vol II: The Implementation.* Addison Wesley, 1994.

[7] Tanenbaum, A.S. Redes de Computadoras, 3ª ed. Prentice Hall Hispanoamericana, 1997.