Curso de Simulación de Redes de Comunicaciones: Planteamiento, Objetivos y Metodología

J. Alcaraz Espín*, E. Egea López, J. García Haro
Departamento de Tecnologías de la Información y Comunicaciones,
Universidad Politécnica de Cartagena.
* Dir.: Cuartel de Antigones, 30202, Cartagena, Murcia.
tlf: +34 968 326 544, fax: +34 968 325 973, e-mail: juan.alcaraz@upct.es

***Resumen*—En este artículo se describe el planteamiento, metodología y contenidos del curso de simulación que se imparte en el currículo de Ingeniero de Telecomunicación en la Universidad Politécnica de Cartagena (UPCT). Los objetivos del programa son que el alumno adquiera conceptos teóricos relacionados con la simulación genérica de sistemas y que sea capaz de emplear las herramientas necesarias para la evaluación por simulación de redes de comunicaciones de datos, es decir, tiene una vertiente práctica y otra teórica. El curso se plantea buscando una mayor implicación del alumno en el proceso de aprendizaje, según las directrices del modelo de convergencia europeo. La metodología está estructurada en dos etapas: una primera de formación y una segunda etapa de desarrollo de un proyecto. En la parte de formación los contenidos se han estructurado en sesiones "autocontenidas", de manera que la práctica pueda ser repetida en cualquier momento y de manera autónoma por el alumno. Además, al alumno se le hace completamente partícipe de los objetivos didácticos. Para ello se le presenta la metodología general en la primera sesión. En cada sesión se le da a conocer de antemano cuáles son los contenidos que debe asimilar y la relevancia de los mismos. En cada experimento, se explica qué concepto se ha tratado de ilustrar. En relación al proyecto, se le dan criterios detallados sobre cómo se va a evaluar, haciendo énfasis en el grado de autonomía, iniciativa y creatividad que hayan demostrado en su desarrollo.**

***Index Terms*—Simulación por Eventos Discretos, OMNeT++, C++, Enseñanza orientada a proyectos, Redes de Ordenadores**

I. Introducción

La simulación por ordenador, como herramienta científica y técnico-profesional, ha ganado importancia progresivamente durante las últimas décadas. Esto es especialmente notable en el ámbito de las redes de comunicaciones, debido por una parte, al aumento de la capacidad de procesado de los ordenadores y, por otra parte, a la creciente complejidad de las redes de comunicación. Esto ha hecho que los modelos matemáticos empleados tradicionalmente para el análisis y diseño de redes de comunicación de datos, fundamentalmente basados en análisis estocástico, vayan dando paso y, en algunos casos, perdiendo terreno frente a las técnicas basadas en simulación.

La formación en estas técnicas está incorporada al currículo de Ingeniero de Telecomunicación en la titulación impartida en la Escuela Técnica Superior de Ingenieros de Telecomunicación (ETSIT) de la Universidad Politécnica de Cartagena (UPCT), dentro de las prácticas de la asignatura Redes de Ordenadores del cuarto curso académico de dicha titulación. En este artículo se describen los objetivos, contenidos, metodología y experiencias de este curso de simulación orientado a redes de comunicaciones de datos. Esta asignatura se implantó durante el curso académico 2002-2003 por el mismo equipo de profesores que ha estado impartiéndola hasta la actualidad.

El objetivo final del curso de simulación es que los alumnos sean capaces de analizar un sistema de comunicaciones y desarrollar un simulador del mismo, basado en eventos discretos, obtener medidas y realizar un análisis estadístico de los resultados. En definitiva, que realicen una evaluación de prestaciones basada en simulación.

Para ello, han de adquirir los conceptos esenciales de simulación por eventos discretos, generación de variables aleatorias y análisis estadístico de resultados. Para el desarrollo del simuladores, se forma al alumno en la librería de simulación por eventos discretos OMNeT++ [2], con la que deberán programar en C++ en entorno Linux. Para una obtención eficiente de resultados los alumnos aprenden a automatizar la toma de medidas y el análisis de resultados con el lenguaje interpretado *perl* [3] y a presentar gráficas de los mismos con *gnuplot* [4]. Estos objetivos se han planteado de manera estructurada en forma de competencias genéricas y específicas a adquirir (ver Sec. II-B), siguiendo las directrices que el modelo de convergencia europea sugiere a la hora de desarrollar proyectos docentes.

Los aspectos a destacar de este curso son, fundamentalmente los siguientes:

- La extensión y diversidad de contenidos del curso, debido a la gran cantidad de competencias a desarrollar en el alumno y de tecnología involucradas. De hecho, no hemos encontrado hasta el momento material docente publicado que de respuesta a los objetivos formativos planteados en este curso.
- El material de laboratorio empleado consiste en *software* de código libre y abierto, para PCs convencionales, lógicamente.
- La metodología, que está estructurada en dos etapas: una primera de formación y una segunda etapa de desarrollo de un proyecto.
  La etapa de formación consta de 10 sesiones de prácticas en las que los alumnos van adquiriendo progresivamente los conocimientos y habilidades técnicas necesarias, mediante introducciones teóricas, experimentos prácticos, cuestiones de autoevaluación, repaso y ejercicios.

La etapa de proyecto consiste en la elaboración de un proyecto relativamente complejo, realizado en grupo y en el que deben implementar un sistema del que se les proporciona una especificación, obtener medidas de rendimiento, analizar e interpretar los resultados y presentarlos, junto con el código, en una memoria completa.

Conviene destacar que los alumnos acceden a la asignatura sin conocimientos previos de C++, aunque sí de programación orientada a objetos. Tampoco han recibido formación en el resto de las herramientas comentadas (OMNeT++, *perl*, *gnuplot*). Esto supone en cierta medida un reto a la hora de plantear la asignatura. En este artículo se explica la estrategia didáctica implementada en estas prácticas, en la cual se dividen los objetivos en dos vertientes: la primera recoge los objetivos de simulación, donde se recogen todos los objetivos relativos a adquirir conocimientos y competencias relativas a las técnicas de simulación (independientemente de la tecnología); la segunda vertiente corresponde a los objetivos denominados “de programación", que engloban los relativos al manejo de las tecnologías y herramientas involucradas. Los objetivos se abordan en paralelo, aunque se pondera la carga de los mismos en cada sesión, de forma que unas sesiones están más centradas en fundamentos teóricos y otras en aspectos de programación.

Se debe tener en cuenta, y así se le hace saber al alumno, que el objetivo de la asignatura no es aprender a programar en C++, por lo que C++ no se aborda de forma exhaustiva, sino muy al contrario, tan sólo se describe la sintaxis básica que el alumno necesitará para implementar la mayoría de los problemas que le pueden surgir durante el transcurso de las prácticas y el desarrollo del proyecto. Para una cobertura más completa, se insta a que el alumno empleé la bibliografía, incentivando su autonomía y su iniciativa a la hora de resolver problemas.

Una parte destacable de la metodología de la asignatura es que al alumno se le hace completamente partícipe de los objetivos didácticos. Para ello se le presenta la metodología general en la primera sesión. En cada sesión se le da a conocer de antemano cuáles son los contenidos que debe asimilar y la relevancia de los mismos. En cada experimento, se explica qué concepto se ha tratado de ilustrar. En relación al proyecto, se le dan criterios detallados sobre cómo se va a evaluar, haciendo énfasis en el grado de autonomía, iniciativa y creatividad que hayan demostrado en su desarrollo.

El resto del artículo está organizado como se explica a continuación. La sección II se presenta el planteamiento de la asignatura, desglosado en el contexto, los objetivos y la metodología general. La sección III se centra en los contenidos, justificando la elección de los mismos y explicando la estrategia de aprendizaje implantada. A continuación, la sección IV aborda el planteamiento de los proyectos. Finalmente, la sección V ofrece las conclusiones extraídas de esta experiencia docente.

II. Planteamiento del curso

En esta sección se proporciona una introducción al planteamiento general de la asignatura. Primero describimos el contexto de la asignatura en el plan de estudios y el perfil de los alumnos. A continuación introducimos los objetivos de la asignatura, así como la metodología que se emplea en su desarrollo.

II-A. Contexto de la asignatura y perfil del alumnado

La asignatura Redes de Ordenadores se imparte en el cuarto de curso de Ingeniero de Telecomunicación en la Universidad Politécnica de Cartagena. Se trata de una asignatura troncal anual, con 7.5 créditos teóricos y 3 créditos prácticos. La asignatura profundiza en conceptos fundamentales ya introducidos en las asignaturas de tercer curso Telemática y Redes y Servicios de Comunicaciones, además de desarrollar específicamente contenidos más avanzados no presentes en otras asignaturas. El objetivo es el de presentar contenidos fundamentales, y lo más permanentes posibles, relacionados con los niveles de red, de transporte y de acceso al medio así como sobre la evaluación de prestaciones en sistemas de comunicaciones a través del análisis y de la simulación.

El perfil de los alumnos que la cursan no es heterogéneo puesto que un importante porcentaje (30 %) de los alumnos matriculados provienen de la especialidad de Telemática de Ingeniería Técnica de Telecomunicación, que pueden acceder sin necesidad de curso puente al segundo ciclo. Por tanto los profesores se encuentran básicamente con dos perfiles de alumno:

- *Alumnos que provienen de Ingeniería de Telecomunicación*. Poseen una base teórica sólida en aspectos del nivel físico y nivel de enlace de datos. Conocen los fundamentos de la programación y nociones básicas de programación orientada a objetos, así como de fundamentos de computadores, pero no de sistemas operativos. Sin embargo, muestran ciertas carencias prácticas: el único lenguaje de programación de propósito general que han empleado es Java, aunque sí dominan ampliamente la programación con Matlab. Se observa que, en general, tienen enormes dificultades para trabajar con el sistema operativo Linux, e incluso muestran cierto rechazo.
- *Alumnos que provienen de Ingeniería Telemática*. Poseen, como es de esperar, conocimientos teóricos en todos los niveles de la pila de protocolos, aunque más superficiales en cuestiones de nivel físico. Al contrario que los alumnos de Ingeniería Superior, tienen amplia experiencia práctica, tanto en manejo y configuración de equipos de comunicaciones, como en lenguajes de programación: Java, C, PHP y otros. Además están acostumbrados a trabajar con Linux y lo manejan con relativa soltura.

En resumen, para los alumnos que procedan del primer ciclo de la Ingeniería Superior, las prácticas de Redes de Ordenadores suponen el primer contacto con el desarrollo de aplicaciones software de cierta complejidad, lo cual puede motivar desaliento en el alumno, factor que es por tanto necesario tener en cuenta y corregir en la medida de lo posible. Por el contrario, los alumnos de la Ingeniería Técnica están más habituados con los problemas prácticos asociados al desarrollo de proyectos de software, pero encuentran mayores dificultades a la hora de trabajar con modelos teóricos.

II-B. Objetivos

Los objetivos del programa de prácticas se han planteado en los términos que propugna el nuevo modelo educativo promovido por el proceso de integración del Espacio Europeo de Enseñanza Superior [1]. De esta forma, los objetivos de la asignatura se han expresado en términos de competencias adquiridas por el alumno. Así, los dos objetivos fundamentales del programa: (1) que el alumno adquiera conceptos teóricos relacionados con la simulación genérica de sistemas y (2) que sea capaz de emplear las herramientas necesarias para la evaluación por simulación de sistemas complejos, se plantean en términos de competencias a adquirir de la siguiente forma:

- *Competencias genéricas*. (1) Capacidad de organización y planificación, (2) capacidad crítica y de trabajo en equipo y (3) capacidad de aprender y de aplicar conocimientos en la practica.
- *Competencias específicas*. Clasificadas en dos tipos: de carácter conceptual (teórico) o procedimental (práctico).
  - *Conceptuales*. El alumno debe conocer: (1) técnicas de generación por computador de números aleatorios, (2) la necesidad de simulación y los distintos modelos de simulación, (3) el algoritmo de simulación por eventos discretos y (4) técnicas de análisis estadístico de resultados de simulación.
  - *Procedimentales*. El alumno debe ser capaz de: (1) programar aplicaciones complejas con C++, (2) implementar un motor de simulación por eventos discretos, (3) utilizar librerías de simulación y consultar la documentación necesaria, (4) manejar herramientas de automatización de las simulaciones, análisis de resultados y presentación de los mismos, (5) dominar comandos básicos del sistema operativo Linux y (6) desarrollar un proyecto de simulación complejo.

Obsérvese que el objetivo fundamental de las prácticas no es el de experimentar los conceptos explicados en la parte teórica de la asignatura, sino proporcionar una visión teórica de los problemas y técnicas de simulación y, sobre todo, una experiencia práctica en la implementación de simulaciones. Aun así, parte de la materia de teoría (sistemas de colas) se utiliza como hilo conductor de las prácticas y el proyecto final de simulación cubre en muchas ocasiones contenidos teóricos de relevancia. Nótese también como los objetivos cubren las dos vertientes del aprendizaje, tanto el "conocer" como el "saber hacer".

II-C. Metodología

La metodología empleada se apoya en dos pilares:

- *Estructuración de los contenidos en sesiones "autocontenidas"*. Cada sesión de prácticas aborda un contenido concreto del temario previsto. Se proporciona un cuadernillo de prácticas para cada sesión. En dicho cuadernillo se incluye, por este orden: (1) los objetivos de la práctica, separados en dos grupos, los relativos a simulación y los relativos a programación; (2) una explicación teórica, más o menos extensa en función del tema tratado, (3) un desarrollo paso a paso muy detallado de las tareas a realizar en la práctica, (4) una sección de revisión de conceptos manejados, (5) una serie de cuestiones cortas de autoevaluación, (6) una conjunto de ejercicios propuestos y (7) bibliografía relacionada.
- *Desarrollo de un proyecto de simulación final*. Se proporciona al alumnado un documento que especifica los requisitos de un proyecto de simulación complejo: (1) introducción teórica, (2) requisitos del simulador, (3) escenarios a simular y (4) resultados a presentar, así como una serie de consejos para el desarrollo y los criterios de evaluación.

La metodología en sí es la siguiente. Dos terceras partes del curso se dedican al desarrollo presencial de las sesiones de prácticas. Los alumnos se agrupan en puestos de dos personas frente al ordenador, que se suelen mantiener a lo largo de todo el curso. En cada sesión, al comienzo el profesor introduce el tema y explica brevemente los contenidos teóricos. En algunos casos, la explicación se apoya en la resolución de uno de los ejercicios planteados en el cuadernillo. Los alumnos deben a continuación seguir el desarrollo de la práctica tal y como se detalla en el cuadernillo y el profesor resuelve las dudas que surgen. Hay que destacar que para cada experimento o tarea que indica el cuadernillo se incluye una explicación del concepto manejado. La sesión finaliza con indicaciones sobre autoevaluación.

El último tercio del curso se dedica al desarrollo del proyecto final. En este caso, el profesor está disponible en el horario habitual de laboratorio aunque la asistencia no es obligatoria. El profesor resuelve dudas sobre el proyecto y proporciona sugerencias e indicaciones respecto a la implementación.

En la evaluación se compone de dos partes: un examen final sobre los contenidos tanto teóricos como prácticos y la evaluación del proyecto final. En el examen se plantean problemas similares a los que se proponen en los cuadernillos de prácticas.

Esta metodología pretende, por una parte, que el alumno esté informado de qué se espera que aprenda en cada práctica y que sea capaz de autoevaluar si se ha logrado el objetivo. Además, la estructura autocontenida y detallada de los cuadernillos permite que la práctica pueda ser repetida en cualquier momento y de manera autónoma por el alumno. De esta forma, se facilita la realización de las prácticas al inevitable conjunto alumnos que tienen problemas para asistir a las mismas por diversos motivos, así como que el alumno pueda trabajar en ellas por su cuenta si considera que no ha cumplido los objetivos que se esperan de él. Se facilita también el estudio de cara al examen final.

Por otra parte, se pretende dotar de flexibilidad al desarrollo de las prácticas. El alumno puede optar por realizar la práctica por su cuenta antes de acudir a la sesión presencial y dedicar ésta a resolver dudas y realizar los ejercicios propuestos; o viceversa. En cualquier caso, el objetivo es que se realice tanto trabajo en clase como en casa.

Finalmente, durante el desarrollo del proyecto final se fomenta que los alumnos se organicen el tiempo y el trabajo a su conveniencia y se habitúen a consultar documentación y apoyarse en bibliografía complementaria.

III. Estructura de los contenidos

En esta sección se exponen de forma razonada los contenidos cubiertos en el curso así como la secuencia con la que estos son presentados al alumno.

III-A. Descripción de los contenidos

Como se comentó en la introducción, los contenidos se organizan en dos grupos claramente diferenciados: relativos a simulación y relativos a programación. Los primeros engloban los conceptos y técnicas relacionadas con el método de simulación por eventos discretos [5], y el análisis estadístico de resultados. Los segundos están dedicados a las herramientas *software* (lenguajes, librerías, programas, etc) que hacen posible que el alumno ponga en práctica los conceptos teóricos de simulación para implementar, depurar y validar simuladores de sistemas de comunicaciones de datos y obtener y representar medidas de rendimiento de los mismos. A continuación se exponen ambos grupos en más detalle.

III-A1. Contenidos relativos a simulación: Los contenidos relativos a simulación cubren conceptos esenciales de análisis numérico, estadística y algorítmica empleados en simulación y análisis de resultados, y que son independientes de las herramientas *software* empleadas. Constituyen la base teórica del curso, y por tanto su comprensión y asimilación por parte del alumno es un objetivo fundamental. Son técnicas y ampliamente empleadas en tareas de diseño, investigación y desarrollo, análisis de prestaciones, etc, y que pueden encontrarse en su totalidad o en parte en múltiples referencias bibliográficas previas, como [5], [7]. A continuación se desglosan en un mayor grado de detalle, mostrando que el rango de contenidos cubierto da lugar a un material didáctico compacto y autocontenido.

1. **Generación de números aleatorios**. La capacidad de generar procesos aleatorios es fundamental a la hora de simular sistemas reales, en los que el comportamiento de las entidades es frecuentemente aleatorio. La generación de estos procesos requiere la generación de variables aleatorias que sigan una determinada distribución estadística, por ejemplo exponencial o normal. La generación de variables aleatorias se realiza en dos pasos: Primero, generación de una secuencia de números aleatorios (n.a.) uniformemente distribuidos entre 0 y 1. Segundo, transformación de la secuencia anterior para producir variables aleatorias (v.a.) que sigan una determinada distribución.
   Para la generación de n.a. se explica la generación de secuencias pseudoaleatorias mediante los generadores congruenciales lineales (GCL), es decir, basados en la siguiente ecuación discreta $x_n = (ax_{n-1} + b)mod(m)$, donde x_n es el valor n-ésimo de la secuencia, a se llama *multiplicador constante*, b es el *incremento*, m el *módulo*. Se estudian las propiedades de los generadores en función de estos parámetros. La referencia bibliográfica más relevante sobre la que se ha elaborado este contenido es [12] y en menor medida [5].
2. **Generación de variables aleatorias**. Las variables aleatorias se generan realizando algún tipo de transformación sobre una secuencia uniforme entre 0 y 1 obtenida mediante un GCL. Los métodos estudiados son:
   - Método de la transformada inversa. Es el método más importante en el ámbito del curso por su utilidad para generar v.a. con distribución exponencial.
   - Método de aceptación rechazo.
   - Generación de variables aleatorias normales mediante la aplicación del Teorema Central del Límite.
3. **Test de hipótesis**. Puesto que las v.a. generadas artificialmente son un elemento fundamental en simulación, se forma a los alumnos en el análisis del grado de fiabilidad de las mismas. Este grado de fiabilidad puede definirse de acuerdo a distintos criterios, por ejemplo: en qué medida la secuencia generada se ajusta a una determinada distribución, ó cuál es el grado de correlación entre muestras consecutivas. En estas prácticas se emplea el test chi-cuadrado, útil para el primero de los dos objetivos de medida mencionados. Como en el apartado anterior, gran parte del contenido tiene como base la referencia [12]. Además de la importancia teórica de dicho método, con su planteamiento se trata de fomentar el pensamiento crítico del alumno.
4. **Simulación por eventos discretos (DES)**. Se trata de una técnica de análisis numérico de gran eficiencia para la simulación de cierto tipo de sistemas, especialmente de aquellos que pueden ser modelados mediante sistemas de espera (colas), como es el caso de las redes de comunicaciones de datos. Si bien es cierto que existen potentes técnicas matemáticas para analizar este tipo de sistemas, su uso está restringido al ámbito de la investigación e, incluso en este ámbito, la creciente complejidad de los sistemas de comunicaciones, hacen en ocasiones imprescindible el uso de simulación, que complementa el análisis estocástico. Es destacable que, dada su eficiencia DES se emplea ampliamente en herramientas de simulación comerciales, como OPNET [10] o de código libre, como ns-2 [11] u OMNeT++ [2]. La formación del alumno en la técnica DES comienza por la comprensión de conceptos esenciales en simulación, como son: sistema, modelo del sistema, entidad, evento (exógeno y endógeno), atributos o propiedades de las entidades, estado del sistema, lista de eventos futuros, actividad y tiempo de simulación. Posteriormente se trata en profundidad el algoritmo en el que se basa DES, denominado *Event-Scheduling/Time-Advance* (que se podría traducir como Programación de Eventos en Tiempo Anticipado). El material de referencia en lo relativo a DES es [5] y en menor medida [6].
5. **Eventos de sistemas de colas**. El análisis de sistemas de colas clásicos, como la cola M/M/1, es objeto de estudio en la parte teórica de la asignatura. Por este motivo se emplea la cola M/M/1 para aprender a definir eventos en un simulador DES. En este sistema sólo existen dos posibles eventos: llegadas de clientes al sistema, y salidas del sistema. La definición de eventos en otros tipos de sistemas (M/M/n, M/M/1/K, M/M/1/K/P, G/G/1, Redes de colas, etc) se aborda en experimentos y ejercicios

propuestos.

6. **Conceptos de verificación y validación de un simulador**. Con el objetivo de fomentar el rigor y el pensamiento crítico del alumnado, se hace especial énfasis en los conceptos de verificación y validación. Se llama verificación de un simulador a la comprobación de la implementación correcta del código. El alumno ha de entender que el hecho de que el código no tenga errores no implica que el modelo se haya implementado correctamente. A la comprobación de la validez del modelo se le llama validación, y en las prácticas de la asignatura se basa en la comparación de resultados de simulación con resultados teóricos.
7. **Estimadores estadísticos**. El alumno debe saber que, en general, el objetivo de simular un sistema de comunicaciones es evaluar algún aspecto relativo al rendimiento del sistema simulado. Una simulación tendrá como resultado conjunto de medidas para cada parámetro evaluado, que han de ser sometidas a un análisis estadístico. En estas prácticas se aplica un análisis sencillo. Para ello se repasan los conceptos de media y varianza, se define el concepto de estimador insesgado y se muestra cómo obtener estimaciones insesgada de la media y de la varianza de un conjunto de medidas.
8. **Muestreo de variables**. Para obtener medidas de los parámetros, es necesario muestrearlos durante el transcurso de la simulación. La implementación de este muestreo depende del tipo de variable muestreada, por lo que el alumno debe aprender a diferenciar y reconocer variables que describan procesos estocásticos en tiempo discreto y procesos estocásticos en tiempo continuo, y saber muestrearlos adecuadamente.
9. **Efecto de los transitorios**. El alumno debe diferenciar entre régimen permanente y régimen transitorio de un sistema, identificar cuándo y porqué el régimen transitorio puede influir en los resultados de las medidas finales en régimen permanente y cómo mitigar o eliminar el efecto de los transitorios.
10. **Distribución normal y *t-student***. En el análisis de resultados de experimentos son de especial importancia dos tipos de distribuciones, la normal y la *t-student*. Aparte de la teoría, el alumno aprende mediante diversos experimentos cuándo es aplicable cada tipo de distribución para caracterizar un resultado de simulación.
11. **Cálculo del intervalo de confianza**. El alumno va descubriendo que tan importante es obtener un estimador adecuado de un parámetro del sistema simulado como saber qué fiabilidad tiene la medida obtenida. Este concepto está también ligado al rigor profesional y al pensamiento crítico. El grado de fiabilidad de una medida lo aporta el intervalo de confianza de dicha medida. Numerosos experimentos, cuestiones y ejercicios ayudan al alumno a dominar el cómputo de intervalos de confianza empleando conjuntos de medidas y la distribución *t-student*.
12. **Factores que influyen en el intervalo de confianza**. El aprendizaje del cálculo de los intervalos de confianza no se limita únicamente a la aplicación de la fórmula matemática que lo define, sino que, por su importancia, se adquiere un conocimiento más profundo del mismo que permita a los alumnos saber en qué medida afecta la configuración del método de medida en el intervalo de confianza. Mediante ejercicios prácticos se comprueba la influencia de el número de experimentos y la duración de los mismos en el valor del intervalo de confianza de una medida.
13. **Métodos de análisis de resultados**. Se definen como tales aquellos métodos que permiten obtener intervalos de confianza. En las prácticas los alumnos aprenden dos métodos esenciales para ello: El método de las réplicas independientes y el método de las medias por lotes. Los métodos se presentan de forma que el alumno conozca los problemas que plantea la medida de parámetros en régimen estacionario y en qué medida cada método puede dar solución a dicho problema. Los problemas identificados son:
    - Sesgo de inicialización en la simulación de estado estacionario, debido a los transitorios.
    - Correlación entre muestras en los sistemas de espera.
    - Obtener una precisión acotada.

III-A2. Contenidos relativos a programación: Este conjunto de contenidos está dedicado al manejo de las herramientas *software* necesarias para poner en práctica todos los conceptos teóricos del bloque anterior, realizar los experimentos didácticos planteados y en última instancia, llevar a cabo el proyecto final en el que los alumnos implementan un simulador de un sistema real, analizar su rendimiento y presentan las medidas obtenidas. Estas herramientas son:

1. **OMNeT++** [2]. OMNeT++ es una librería de simulación por eventos discretos, es decir, proporciona la funcionalidad común a este tipo de simuladores: una implementación de la FEL, un planificador, un reloj de simulación y una extensa librería de utilidades. El objetivo es que el programador se centre en la descripción del modelo fundamentalmente, sin preocuparse de la implementación del núcleo de simulación.
   OMNeT++ está implementado en C++, es decir, es una librería de clases C++. Se utiliza un modelo de programación basado en componentes, es decir, los modelos en OMNeT++ tienen una estructura modular: hay entidades simples, que realizan las actividades. Estos módulos simples se incluyen en módulos complejos que se componen de varios módulos simples y que van formando una estructura anidada jerárquica. Los módulos se comunican entre sí mediante intercambio de mensajes, de forma que un evento en OMNeT++ es la llegada de un mensaje a un módulo. Además OMNeT++ proporciona un entorno gráfico de simulación que nos permite depurar y validar más fácilmente el simulador.
   Todas estas características, y el hecho de que OMNeT++ está distribuido como código libre, hacen de OMNeT++ una herramienta adecuada para la docencia. Otra

posible opción es ns-2, también de código libre. Ésta librería tiene la ventaja de que está más extendida en el ámbito científico y cuenta, por tanto, con una comunidad de desarrolladores mucho más amplia. Sin embargo su complejidad es mayor y no aporta una interfaz gráfica tan completa e intuitiva como OMNeT++. Esto hace que la curva de aprendizaje de ns-2 sea más lenta que OMNeT++, lo que hace a este último una opción mejor para un curso de 30 horas. Por otra parte, OMNeT++ cuenta con una comunidad creciente de usuarios, y su presencia en el ámbito científico es cada vez mayor.El área de Ingeniería Telemática que imparte esta asignatura en la UPCT forma parte activa de esta comunidad de usuarios, como demuestra la participación reciente en el primer congreso internacional de OMNeT++ [13], y otras publicaciones internacionales relativas a metodología de simulación [14], [15] , en donde participan los profesores de la asignatura.

2. **C++**. Éste es el lenguaje en el que están programadas las clases que componen la librería C++. Puesto que el proceso de desarrollo de un simulador consiste en crear clases derivadas de las clases proporcionadas por OMNeT++, dándoles funcionalidad, el alumno debe ser capaz de programar código en C++.
   El nivel de competencia necesario en C++ no es muy elevado, teniendo en cuenta que la programación de módulos en OMNeT++ consiste fundamentalmente en emplear los métodos de las clases predefinidas. Sin embargo sí que es necesario que el alumno domine algunos aspectos esenciales para poder desarrollar modelos de simulación de cierta complejidad. Algunos de estos aspectos de C++ son:
   - Tipos básicos de datos.
   - Métodos de entrada y salida.
   - Declarar, definir e instanciar clases.
   - Invocación de métodos de objetos. Paso de variables.
   - Bucles y estructuras de control.
   - Herencia y polimorfismo en C++.
   - Manejo de plantillas de contenedores de la *Standard Template Library* (STL).

   Las primeras sesiones de prácticas contienen material introductorio a C++, adaptado de algunas de las referencias bibliográficas que se proporcionan al alumno, siendo las de mayor relevancia por su profundidad de contenidos y calidad pedagógica, las referencias [8] y [9], de acuerdo al criterio de los profesores.
3. **Linux**. La elaboración, compilación y ejecución de programas en C++ resulta especialmente sencillo en Linux, empleando herramientas de libre distribución, siendo Linux, en sí mismo, un sistema operativo de código libre y abierto. Además, las herramientas que incluyen las distribuciones también facilitan enormemente la automatización de tareas (*perl*) y la presentación de resultados (*gnuplot*). Los alumnos han de adquirir en Linux competencias a nivel de usuario.
4. ***perl***. *Perl* [3] es un lenguaje interpretado de alto nivel que permite automatizar tareas, como ejecutar programas, leer y escribir en ficheros, etc. Los alumnos deben utilizarlo para configurar y ejecutar automáticamente múltiples simulaciones, procesar las medidas realizadas en cada simulación y presentar los resultados de forma que puedan extraer conclusiones útiles sobre los mismos. Dada la naturaleza del método de las réplicas independientes, la automatización de simulaciones es la técnica más eficiente para obtener los intervalos de confianza de las medidas.
5. ***gnuplot***. Se trata de un programa para generar gráficas de gran calidad, tanto en 2 como en 3 dimensiones, a partir de funciones matemáticas o ficheros de datos. Este programa está ampliamente extendido en la comunidad científica por la calidad de los gráficos que genera, que pueden además exportarse a distintos formatos de fichero. Los alumnos emplean *gnuplot* durante todo el curso para realizar gráficas relacionadas con la simulación. Entre las diversas funcionalidades que emplean destacan:
   - Generar gráficas a partir de ficheros de datos.
   - Modificar los tipos de línea y de punto.
   - Incluir títulos y leyendas en las figuras.
   - Imprimir gráficas con varias curva.
   - Representar los intervalos de confianza.
   - Representar histogramas.

III-B. Secuencia temporal

La secuencia temporal de los contenidos ha sido planificada y depurada a lo largo de los años para ajustarlos el ritmo de presentación al ritmo de asimilación adecuado al alumno. En este sentido es especialmente importante destacar que las sesiones no contienen un reparto equitativo en contenidos, ni en cantidad total ni en la proporción de contenidos de simulación y contenidos de programación. En el diagrama de la figura 1 se muestra una secuencia temporal en la que observamos, en cada sesión qué conjunto de contenidos se trabajan.

IV. Proyectos desarrollados

Cada curso se propone un proyecto de simulación que aborda el estudio de un problema clásico en telemática. En general, se intenta buscar proyectos que permitan una amplia variedad de modificaciones con mínimas variaciones en el código para tener flexibilidad a la hora de plantear requisitos mínimos y funcionalidad adicional y de los resultados a exigir. En varios de los trabajos se exige además la validación del simulador sin indicar directamente cómo, de manera que los alumnos deben buscar un modelo teórico y un experimento adecuado.

Con los proyectos se persiguen varios objetivos: Que los alumnos pongan en práctica los conceptos teóricos de desarrollo de modelos de simulación y apliquen y desarrollen las competencias adquiridas en el manejo de herramientas

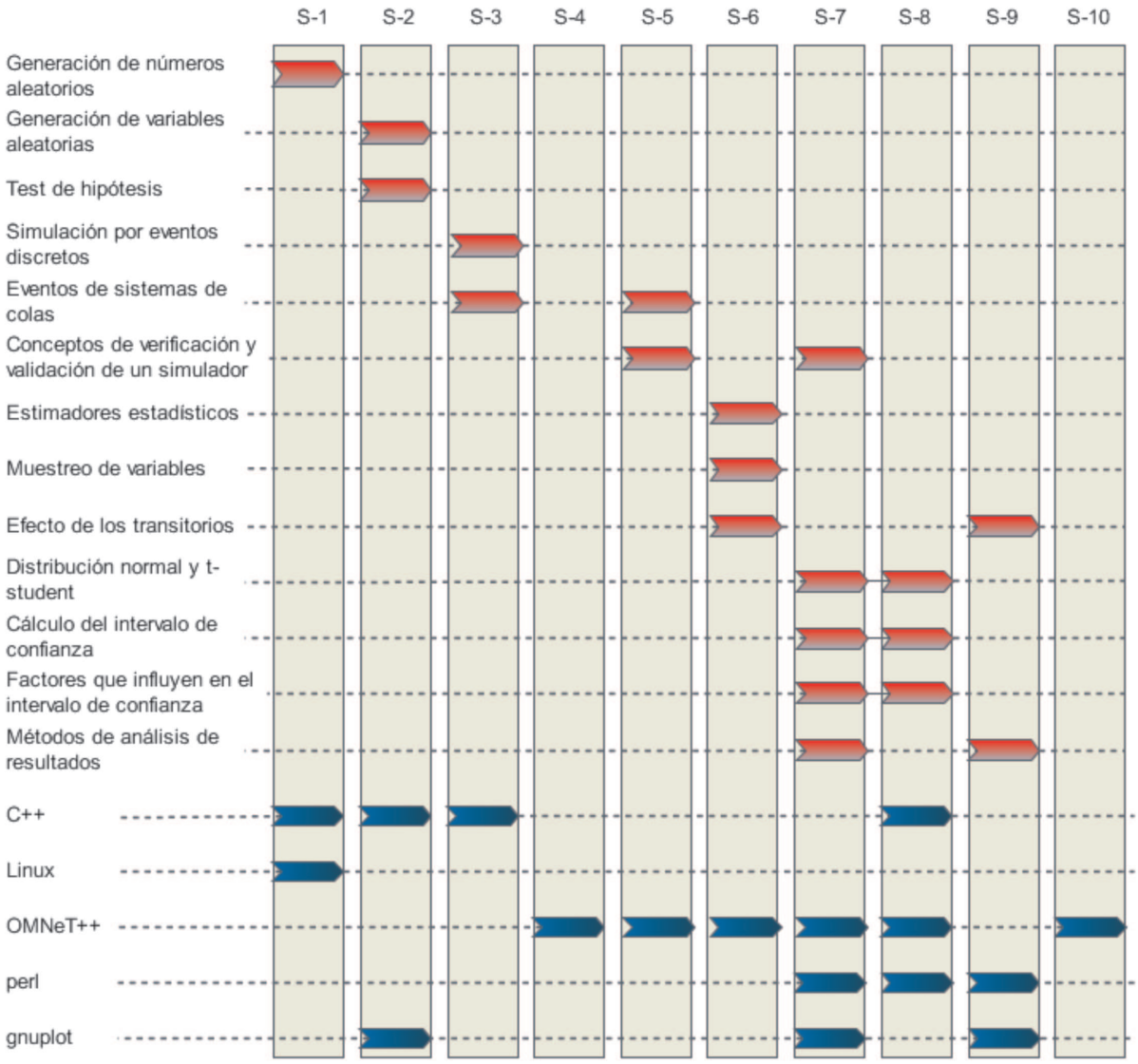

Figura 1. Secuencia temporal de contenidos por sesiones. Las líneas rojas denotan contenidos de simulación y las azules de programación.

software. Al mismo tiempo, se les plantean trabajos relacionados con redes reales, tratando de despertar su curiosidad e interés y favoreciendo un conocimiento más profundo de dichas tecnologías. El planteamiento del proyecto contiene partes con una especificación más cerrada, lo que por un lado evita complicar en exceso la tarea asignada y por otro les fuerza a trabajar con el rigor y el orden necesarios en el diseño e implementación de especificaciones de sistemas de comunicaciones. Otras partes de la propuesta se dejan más abiertas para fomentar la iniciativa y la creatividad de los alumnos, así como su capacidad en la resolución de problemas. El trabajo requiere también que los alumnos trabajen en grupo, con todo lo que ello implica, y desarrollen también sus capacidades de comunicación escrita en la redacción de la memoria, de la que se le proporciona una guía general de elaboración y unos criterios de evaluación muy claros.

Resumimos a continuación los proyectos planteados hasta el momento:

- *Ethernet*. Se estudia el control de acceso al medio y el protocolo CSMA/CD, mediante la simulación de una red Ethernet. El sistema a implementar consta de N estaciones y un concentrador. No se proporciona ningún código al alumno, debe partir de cero. La implementación tiene un nivel de complejidad medio y permite una gran flexibilidad: se puede plantear incrementalmente, empezando por un CSMA simple, añadiendo persistencia, etc. Una vez desarrollado la estructura básica, se pueden probar multitud de algoritmos de ajuste de la ventana de contienda, con mínimas modificaciones del código. Además, parametrizando adecuadamente la solución, permite también una gran flexibilidad a la hora de experimentar la influencia en el rendimiento de diversos parámetros en el protocolo: la distancia entre estaciones, el tamaño de trama, la velocidad de transmisión, el tamaño de

la ventana de contienda, el algoritmo de *back off*, por citar sólo algunos. Como resultados se piden gráficas de distintos parámetros frente a la carga ofrecida: caudal, retardo medio del paquete, retardo medio de acceso al canal, etc.

- *TCP*. El objetivo de este proyecto es programar los mecanismos de control de congestión del protocolo *Transport Control Protocol* (TCP), el más empleado a nivel de transporte en *Internet*. Concretamente los alumnos implementaron el mecanismo especificado en [18], conocido como *Fast Retransmit Fast Recovery*, así como el cómputo del *Round Trip Time* (RTT) y el *Retransmission Time Out* (RTO), especificados en [17]. El simulador puede ser validado mediante la inspección de las trazas de diversos parámetros de TCP, y permite a los alumnos experimentar con fuentes de datos TCP en situaciones de congestión. Se proponen diversos escenarios en los que los alumnos comprobarán el efecto del ancho de banda disponible, el tamaño del *buffer* en el *router*, y el retardo en el rendimiento de las fuentes, caracterizado por la tasa de recepción de paquetes, la tasa de pérdida de paquetes y el retardo e transferencia. También se comprueban otros efectos como el reparto del ancho de banda entre distintas fuentes o la sincronización de las fuentes. De esta forma el alumno va descubriendo por sí mismo problemas clásicos en el tráfico de datos, lo que constituye la base para la explicación de tecnologías más recientes que dan respuesta a dichos problemas, como la gestión activa del *buffer*, los mecanismos de calidad de servicio o las versiones más recientes de TCP.
- *OSPF*. Se estudia el encaminamiento basado en estado del enlace mediante el protocolo OSPF (*Open Shortest Path First*) [19]. La implementación completa del protocolo se considera demasiado compleja, así que se propone una versión con funcionalidad reducida del mismo. Para el cálculo de la mejor ruta se propone la implementación del algoritmo de Dijkstra. Este sistema también ofrece mucha flexibilidad, puesto que se pueden probar diferentes algoritmos de búsqueda de la mejor ruta y múltiples métricas para los enlaces. Cabe destacar que por la relación tan directa con los contenidos de teoría, se pueden plantear una variedad de estudios teóricos relacionados con el tema. De hecho, varias de las soluciones entregadas los incluían incluso aunque no se exigieran como parte del proyecto.
- *ATM*. El proyecto consiste en que los alumnos programen el mecanismo de *Control del Parámetro de Utilización* UPC del Modo de Transferencia Asíncrono (ATM). El UPC consiste en un doble mecanismo de *Leacky Bucket* [16], y se emplea en el simulador para gestionar el tráfico proveniente de unas fuentes de tasa variable cuyo código se proporciona. El simulador permite a los alumnos comprobar el efecto de la multiplexación estadística en ATM, entender los descriptores de tráfico y cómo influyen en la tasa de pérdida de celdas de las fuentes. Estos experimentos permiten entender cómo las fuentes pueden adecuar su tráfico mediante la técnica de *Traffic Shaping*, que los alumnos implementan mediante un sencillo mecanismo de *token bucket*. Los alumnos comprueban que los flujos de datos mejoran su calidad de servicio (o al menos algunos parámetros de calidad) sin aumentar el ancho de banda de la red. Otra técnica, muy sencilla de implementar para los alumnos y que logra interesantes resultados en este sentido es el descarte selectivo de celdas.
- *Algoritmos ARQ*. Se aborda el estudio de los algoritmos ARQ clásicos: *Go-Back-N* y *Selective Repeat* [16]. En este proyecto, puesto que consideramos que la complejidad de la implementación del simulador no es muy alta, se hace especial hincapié en la automatización de los resultados obtenidos, es decir, se exige la presentación de gráficas variando múltiples parámetros.

V. Conclusiones

En este artículo se ha descrito el planteamiento, metodología y contenidos de las prácticas de las asignatura Redes de Ordenadores, que consisten en un curso de simulación y programación. Desde el momento en el que se implantó el curso en el año 2002, el planteamiento del mismo busca una implicación mayor del alumno en el proceso de aprendizaje, de acuerdo a las directrices del nuevo modelo del EEES. El formato del curso actualmente es el resultado del proceso iterativo de evaluación y realimentación obtenido cada año que se ha impartido.

La experiencia hasta el momento nos ha permitido extraer dos conclusiones relevantes. Por una parte, la percepción general de los alumnos es que el curso es duro y requiere un esfuerzo considerable. En nuestra opinión, esta percepción viene motivada por dos factores: uno particular para el caso de la UPCT, que es la falta de experiencia en el desarrollo *software* de los alumnos de la Ingeniería Superior. El segundo, de carácter general, ya advertido en literatura previa [20], es la dificultad de adaptación a un modelo en el que se exige mucho más trabajo diario e iniciativa personal que el modelo clásico, que en cierto modo prima la pasividad durante el curso y el esfuerzo final para la preparación de los exámenes. De hecho, se ha observado el mismo problema en otros cursos ya adaptados al EEES que se han empezado a impartir este año. Sin embargo la calidad de los proyectos finales de simulación presentados año tras año es, por lo general, muy alta. La conclusión que extraemos de este hecho es que, a pesar de la resistencia inicial de los alumnos ante la relativamente elevada carga de trabajo y de contenidos del curso, la estrategia empleada consigue los objetivos de aprendizaje planteados.

Agradecimientos

Este trabajo ha sido financiado parcialmente por el proyecto TEC2007-67966-01/TCM (CON-PARTE-1) y también se ha desarrollado en el ámbito del “Programa de Ayudas a Grupos de Excelencia de la Región de Murcia, Fundación Seneca, Agencia de Ciencia y Tecnología de la RM (Plan Regional de Ciencia y Tecnología 2007/2010)".

REFERENCIAS

[1] C. Camiña Catalá, J. M. Martínez Rubio, E. Ballester Sarriás, *¿EC Qué?: De la formación en conocimientos a la formación en competencias*, Actas del *XII Congreso Univesitario de Innovación Educativa en las Enseñanzas Técnicas*, Barcelona, 2004.

[2] A. Varga, *OMNeT++ Discrete Simulation System*. URL: *www.omnetpp.org*.

[3] L.Wall, T. Christiansen y J. Orwant, *Programming Perl*, 3 ed., O'Reilly Media, Inc., 2000.

[4] URL: *www.gnuplot.info*.

[5] J. Banks, *Discrete-Event Systems Simulation*, 3 ed., Prentice Hall International, 2000.

[6] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling* Wiley- Interscience, 1991.

[7] J.J. Pazos, A. Suárez y R.P. Díaz, *Teoría de Colas y Simulación por Eventos Discretos*, Prentice Hall, 2003.

[8] B. Eckel, *Thinking in C++, vol. 1*, 2 ed., Prentice Hall, 2000.

[9] B. Eckel, *Thinking in C++, vol. 1*, 2 ed., Prentice Hall, 2003.

[10] OPNET Technologies Incorporated, URL: *www.opnet.com*

[11] The Network Simulator ns-2, URL: *www.isi.edu/nsnam/ns/*

[12] D. Knuth, *The Art of Computer Programming, Vol. 2, Seminumerical Algorithms*, Addison-Wesley, 1997.

[13] J. Alcaraz, G. Pedreño, F. Cerdán y J. García-Haro, *Simulation of 3G DCHs Supporting TCP Traffic: Design, Experiments and Insights on Parameter Tuning*, Actas del *1st International Workshop on OMNeT++*, Marsella, Marzo 2008.

[14] E. Egea López, F. Ponce Marín, J. Vales Alonso, A. Santos Martínez Sala, J. García Haro, *OBIWAN: wireless sensor networks with OMNET++*, *13th IEEE Mediterranean Electrotechnical Conference*, Málaga, Junio 2006.

[15] E. Egea López, J. Vales Alonso, A. Santos Martínez Sala, P. Pavón Mariño, J. García Haro, *Simulation scalability issues in wireless sensor networks*, *Communications Magazine*, vol. 44-7, pp. 64-73, 2006.

[16] W. Stalling, *Redes e Internet de Alta Velocidad. Rendimiento y Calidad de Servicio*, Pearson Prentice Hall, 2004.

[17] V. Paxson y M. Allman, *Computing TCP's Retransmission Timer*, RFC 2988, *Internet Engineering Task Force*, URL: www.ietf.org, 2000.

[18] M. Allman, V. Paxson y W. Stevens, *TCP Congestion Control*, RFC 2581, *Internet Engineering Task Force*, URL: www.ietf.org, 2004.

[19] J. Moy, *OSPF Version 2*, RFC 2328, *Internet Engineering Task Force*, URL: www.ietf.org, 1998.

[20] J.M. Martínez, C. Camiña, E. Ballester, *Reforma metodológica en la enseñanza de Electrónica e Informática Industrial en la UPV*, Actas del *Symp. de Innovación Universitaria "Diseño, desarollo y evaluación del curriculum univesitario"*, Barcelona, 1995.