XXXVIII Jornadas de Automática
https://doi.org/10.17979/spudc.9788497497749.0522



# Aprendiendo Simulación de Eventos Discretos con JaamSim

Enrique Teruel
Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, eteruel@unizar.es

Rosario Aragüés
Departamento de Informática e Ingeniería de Sistemas, Universidad de Zaragoza, raragues@unizar.es

## Resumen

*El aprendizaje, en particular de materias técnicas como la simulación de sistemas dinámicos, requiere que los estudiantes desarrollen casos de diverso nivel de complejidad, desde pequeños ejemplos didácticos hasta proyectos de cierta envergadura. Para ello es ineludible el uso de herramientas profesionales accesibles, lo que es prácticamente sinónimo de código abierto. En este artículo informamos de nuestra experiencia con JaamSim, un paquete de simulación que incluye una interfaz "arrastrar-y-soltar", gráficos interactivos, procesamiento de entradas y salidas, y herramientas de desarrollo de modelos. El grado de madurez de la herramienta, y su comunidad de usuarios, nos parece más que suficiente para sustituir con ventaja otras opciones propietarias o con licencias de estudiante limitadas, si bien JaamSim todavía debe seguir evolucionando, sobre todo en aspectos de usabilidad, para lo que la contribución de los usuarios es fundamental. Una ventaja de algunas populares herramientas propietarias es la existencia de material didáctico, pero consideramos que puede suplirse ventajosamente compartiendo en abierto material didáctico análogo, especialmente ejemplos y casos desarrollados, por lo que contribuimos con el material de nuestro curso, que esperamos aumentar y perfeccionar en sucesivas ediciones.*

**Palabras Clave**: Simuladores, Lenguajes de simulación, Sistemas de eventos discretos, Educación, Herramientas software.

## 1 INTRODUCCIÓN

A la hora de diseñar una asignatura, y sus procesos de enseñanza-aprendizaje, nos parece primordial centrarse en el trabajo que harán los estudiantes en las sesiones prácticas y por su cuenta, pues como mejor se aprende es haciendo (Brown *et al.*, 2014), poniendo en práctica los conceptos explicados y usando herramientas de nivel profesional. Esto es más cierto si cabe cuando la modalidad de enseñanza no es plenamente presencial, y los estudiantes deben realizar su aprendizaje práctico autónomamente, algo que se ha revelado de forma acuciante tras la aparición de cursos masivos en línea (MOOC's), donde predominan las versiones de código abierto como herramientas para todo tipo de materias, especialmente aquéllas en las que, al igual que en la simulación, se requiere ponerse manos a la obra para aprender y compartir (Conole, 2013).

Este era el reto al que nos enfrentábamos al rediseñar la asignatura de Simulación de Sistemas Dinámicos en el grado de Ingeniería Electrónica y Automática de la Universidad de Zaragoza, que se imparte en la Escuela de Ingeniería y Arquitectura, en Zaragoza, y en la Escuela Universitaria Politécnica de Teruel, donde se está planteando la conveniencia de ofrecer el curso en modalidad semipresencial. Previamente la asignatura se impartía utilizando Arena para la simulación de eventos discretos y Matlab-Simulink para la simulación de sistemas continuos e híbridos. Desgraciadamente, estas herramientas sólo estaban disponibles en los laboratorios, pues ambas tienen costes de licencia elevados. En realidad de la primera hay una versión para estudiantes gratuita pero, como suele suceder en estos casos, está drásticamente limitada. Tras un análisis de alternativas más accesibles, idealmente de código abierto, nos decantamos por sustituirlas con JaamSim y OpenModelica, respectivamente. En este artículo nos centramos en el caso de la simulación de eventos discretos, con JaamSim (JaamSim Development Team, 2017), pues la disponibilidad de herramientas de código abierto para simulación de eventos discretos es relativamente reciente y todavía escasa (Dagkakis y Heavey, 2016), mientras que el uso de Modelica (Mattson *et al.*, 1998), un lenguaje estándar de modelado, abierto y con otras ventajas fundamentales para el modelado de sistemas dinámicos, consideramos que está más extendido y documentado (Fritzson, 2015; Tiller, 2014; Martín *et al.*, 2005). No obstante, todo el material del curso se deja accesible en abierto, también el cubierto por OpenModelica (herramienta de código abierto basada en Modelica), que consiste en un "OpenModelica Notebook" adaptado de "DrModelica", basado en (Fritzson, 2015), incluyendo numerosos ejemplos de (Tiller, 2014), y con algunas contribuciones originales.

El material en abierto, accesible a través de (Teruel, 2017), de libre distribución (bajo licencia CC-BY-SA), es prácticamente el mismo que se pone a disposición de los estudiantes, y consiste en presentaciones y otros documentos y archivos compartidos en Google Drive, incluyendo ejemplos realizados con las herramientas. Aunque sea muy brevemente, queremos destacar la conveniencia de la plataforma Google Drive (en nuestro caso dentro del programa Google Apps for Education, suscrito por nuestra Universidad, y combinada con nuestra plataforma LMS, que está basada en Moodle). Los estudiantes pueden acceder a las presentaciones completas, con sus animaciones y anotaciones, y enlaces al material complementario desarrollado (modelos, hojas de cálculo, etc), y para los profesores supone un considerable ahorro de tiempo a la hora de editar y compartir el material.

Tras una primera edición del curso consideramos que nuestros objetivos se han cumplido muy satisfactoriamente, lo que nos ha motivado a poner en común la experiencia para que pueda ser reutilizada y enriquecida por nuestros colegas. Pensamos, además, que muchas de las ideas metodológicas y logísticas pueden aprovecharse igualmente en otras disciplinas, en particular las hemos puesto también en práctica en una asignatura básica de Sistemas Automáticos (en el grado de Ingeniería Mecánica) donde nos hemos apoyado en Scilab/Xcos como alternativa a Matlab-Simulink.

# 2 DISEÑO DEL PROGRAMA DE SIMULACIÓN DE EVENTOS DISCRETOS

Tras una introducción general a la simulación, que podría definirse como experimentación con modelos (típicamente, aunque no exclusivamente, matemáticos-informáticos), el curso se divide en dos partes, una dedicada a los sistemas de eventos discretos y otra a los sistemas continuos e híbridos. Dado que el contexto de la asignatura es un grado de ingeniería electrónica y automática, el énfasis se pone en sistemas de producción y logísticos, y en sistemas técnicos automatizados, aunque esta disciplina interesa a otros perfiles, y de hecho se imparte en numerosas universidades también a estudiantes de la rama económica-empresarial, de la rama informática, o de la rama científica.

## 2.1 ¿LENGUAJE DE PROPÓSITO GENERAL, LENGUAJE DE SIMULACIÓN, O SIMULADOR?

El perfil de los estudiantes motiva una de las primeras decisiones a tomar en el diseño: usar como herramienta o vehículo un lenguaje de programación de propósito general, un lenguaje de simulación, o un simulador. Es un debate que se ha recogido en la literatura (Leemis y Park, 2006), y en el que, a nuestro juicio, la respuesta más apropiada es diferente para cada perfil: mientras para los estudiantes informáticos, o para los más interesados en construir simuladores específicos, la opción preferente será un lenguaje de propósito general, para los estudiantes con perfil de gestión es preferible un simulador capaz y amigable. En nuestro caso, entre ambos extremos, pensamos que deben mostrarse las bases para la programación de simuladores, pero conviene usar un lenguaje de simulación de alto nivel que permita desarrollar y analizar desde el principio ejemplos gradualmente más complejos, punto de vista que coincide con (Law y Kelton, 2000), que es uno de los libros seleccionados como referencia básica. Sin embargo, a diferencia de (Kelton *et al.*, 2015), preferimos una herramienta de código abierto, de forma que los estudiantes no vean limitada su capacidad de modelado (cosa que sucede en la versión para estudiantes de Arena), y a la vez tengan la posibilidad de acceder al código fuente, para adaptarlo o para aprender sobre la construcción de un simulador profesional.

## 2.2 BIBLIOGRAFÍA Y HERRAMIENTA RECOMENDADA

La decisión sobre el tipo de herramientas a usar para construir modelos de simulación condiciona la bibliografía básica a recomendar, que en nuestro caso han sido los libros de Law y Kelton (2000) y de Altiok y Melamed (2007), este último por consideraciones de disponibilidad para nuestros estudiantes (nuestra biblioteca dispone de la versión electrónica, luego todos los estudiantes pueden disponer de él), a falta de un buen texto con acceso abierto, o/y de un texto que se apoyase en JaamSim.

El programa resultante (sobre simulación de eventos discretos) ha sido:

- Introducción a la simulación de eventos discretos
- Lenguajes de simulación y simuladores. Introducción a JaamSim
- Modelado y simulación de sistemas de producción y logísticos
- Técnicas estadísticas
- Desarrollo de casos

Con algo de inspiración “flipped-classroom” nuestra intención era dedicar las clases (unas veinte horas para esta parte) a presentar e ilustrar los conceptos siempre sobre ejemplos lo bastante “ricos”, de forma activa y participativa, y no para explicaciones que pueden, o deben, leerse/verse fuera de clase.

Esto conduce a la tercera decisión, que fue seleccionar y desarrollar en JaamSim ejemplos, en general tomados de los textos de referencia, y dejarlos a disposición de los estudiantes, tanto por el

interés conceptual que tienen los sistemas modelados y analizados como por la utilidad para aprender por imitación/inspiración las técnicas básicas de modelado.

## 2.3 UN REPOSITORIO DE EJEMPLOS

Nuestra intención es desarrollar y ampliar un conjunto variado de ejemplos, bien documentados. Probablemente incorporaremos el resultado de los mejores trabajos de nuestros estudiantes. Inicialmente se han desarrollado los siguientes ejemplos, adaptados a partir de la literatura recomendada, donde se discuten o ilustran con otras herramientas, típicamente Arena:

- Un ejemplo introductorio básico (servidor-cola), adaptado de la Sección 5.2 de (Altiok y Melamed, 2007) y la Sección 3 del manual de usuario de JaamSim (JaamSim Development Team, 2017) que se desarrolla paso a paso a la vez que se presentan conceptos fundamentales sobre modelado y análisis de simulaciones de Montecarlo.
- Una línea de producción, adaptado del Cap. 11 de (Altiok y Melamed, 2007).
- Un sistema de fabricación flexible con AGV's para transporte entre células, adaptado de la Sec. 13.5 de (Law y Kelton, 2000).
- Un ejemplo sobre políticas Push, Pull y Kanban, inspirado en el ejemplo "Kanban Game. Making Robots" de la colección de The Big Lean Simulation Library (JaamSim Development Team, 2017).
- Dos ejemplos de cadena de suministro, adaptados del Cap. 12 de (Altiok y Melamed, 2007).
- Un ejemplo de un peaje, adaptado de la Sec. 13.4 de (Altiok y Melamed, 2007).

Estos ejemplos se usan en clase para presentar los conceptos a la vez que se enseña a modelar y analizar, y varios se trabajan en las clases prácticas, cuyo contenido puede verse en el documento "Cuestionarios" de (Teruel, 2017), en tareas de modelado, depuración de errores y verificación o de análisis y optimización de prestaciones. También sirven como inspiración para los estudiantes en el momento de realizar sus trabajos de asignatura, que constituyen la principal actividad evaluable, y que este primer curso hemos tomado de los "Arena Contest Problems" de (Kelton *et al.*, 2015).

Las primeras versiones de algunos de estos ejemplos han sido compartidas en la comunidad de usuarios de JaamSim, donde hemos sugerido que se centralice la puesta en común de experiencias docentes (tema que se fijó como cabecera) y en particular que se favorezca y ordene la compartición de ejemplos. Los ejemplos que hemos compartido en la comunidad han resultado indudablemente de interés. En concreto, el ejemplo de un sistema de fabricación flexible, del que se habla más adelante, ya ha sido recomendado por algunos usuarios para responder a preguntas de otros sobre el modelado de sistemas análogos.

Dado que los estudiantes desarrollan sus propios ejemplos en las clases prácticas y, sobre todo, en los trabajos, nuestra intención es ir aumentando este repositorio inicial, y mejorar su documentación. Esto permitirá que en sucesivas ediciones del curso, en lugar de "partir de cero", algunos estudiantes puedan profundizar o sofisticar casos disponibles, que es una situación con la que frecuentemente se encontrarán en su actividad profesional.

## 2.4 EJEMPLO DE EJEMPLO

A modo de ejemplo de lo que contiene un ejemplo, tomamos el Sistema de Fabricación Flexible con AGV's. Consta de cinco estaciones de trabajo, cada una con varias máquinas idénticas, a través de las cuales pasan los productos, de diferentes tipos (con diferentes planes de trabajo). El movimiento entre unas estaciones y otras lo hace uno o varios AGV's, y se consideran las alternativas de servicio al siguiente (FIFO) o al más próximo (Nearest). En la carpeta del ejemplo (Teruel, 2017) se encuentra una presentación, la versión de JaamSim con la que se desarrolló, varios modelos con distintas variantes, y archivos y hojas de cálculo con los datos de salida y su análisis. La Figura 1 muestra un pantallazo de dicha carpeta, una hoja de cálculo con resultados, y el modelo, en el que no se ha pretendido hacer una representación "física" del lay-out, sino "funcional": las estaciones se modelan mediante entidades Resource, los tiempos de proceso mediante entidades Delay, y los AGV's mediante entidades de tipo Container que "cargan" una entidad que modela un producto y atraviesan dos entidades Delay, una modelando el tiempo para ir a recoger el producto y otra para modelar el tiempo de llevarlo a su destino. En la Figura 1 tenemos, por ejemplo, tres productos siendo procesados por sendas máquinas en la primera estación, mientras uno espera a que quede una máquina libre. En la estación 3 las cinco máquinas están ocupadas, y tiene otros cuantos esperando (la evolución de la longitud de las colas de las estaciones 2 y 3 se representa dinámicamente en el gráfico de la derecha). Los dos AGV's están ocupados, uno yendo a buscar un producto y otro transportando otro producto a la estación que le corresponde. Los distintos tiempos de proceso y transporte y los detalles de la lógica del modelo están especificados en las diversas distribuciones, asignaciones de atributos, evaluación de expresiones, etc, que normalmente se ocultarían en un modelo de animación, pero en este caso está todo a la vista, pues de lo que se trata es de revisar cómo está construido el modelo, y luego hacer las simulaciones necesarias para analizar y optimizar el sistema. Para facilitar el

análisis en sus primeras fases es muy conveniente la posibilidad de mostrar en la pantalla de animación cuadros de texto y gráficos con los valores de las medidas de prestaciones de interés, en este caso los tiempos de espera (lead times) y sus diversas componentes, la productividad (throughput), la longitud de las colas, las utilizaciones, etc.

## 3 VALORACIÓN DE JaamSim

Tras una exploración de herramientas disponibles, tanto propietarias como sobre todo de acceso abierto (Dagkakis y Heavey, 2016), seleccionamos JaamSim por tratarse de una aplicación relativamente madura, potente y usable, que se deriva de una amplia experiencia, desde 2002, en proyectos industriales de simulación programada directamente en Java (King y Harrison, 2010). A partir de esta experiencia, sus desarrolladores empezaron a construir una herramienta que pudiera ser alternativa a los paquetes comerciales (King y Harrison, 2013). Al principio los propios autores consideraban JaamSim especialmente orientada a programadores, que encontrarían ventajas programando en Java y usando JaamSim frente al uso del lenguaje propio de aplicaciones, cuando éste se requería extensivamente, pero reconocían que era todavía insuficientemente versátil y usable para usuarios no programadores. A nuestro juicio, a la vista de la trayectoria seguida desde que la empezamos a usar en 2016, la aplicación ya ha alcanzado, o está alcanzando, el nivel de desarrollo y usabilidad que la hace atractiva también como paquete de simulación alternativo a los comerciales, para usuarios de todo tipo, y ha seguido y sigue mejorando en este sentido.

Siguiendo las recomendaciones de (Law y Kelton, 2000), los puntos a valorar en un paquete de simulación son los siguientes:

- Capacidades generales, incluyendo flexibilidad

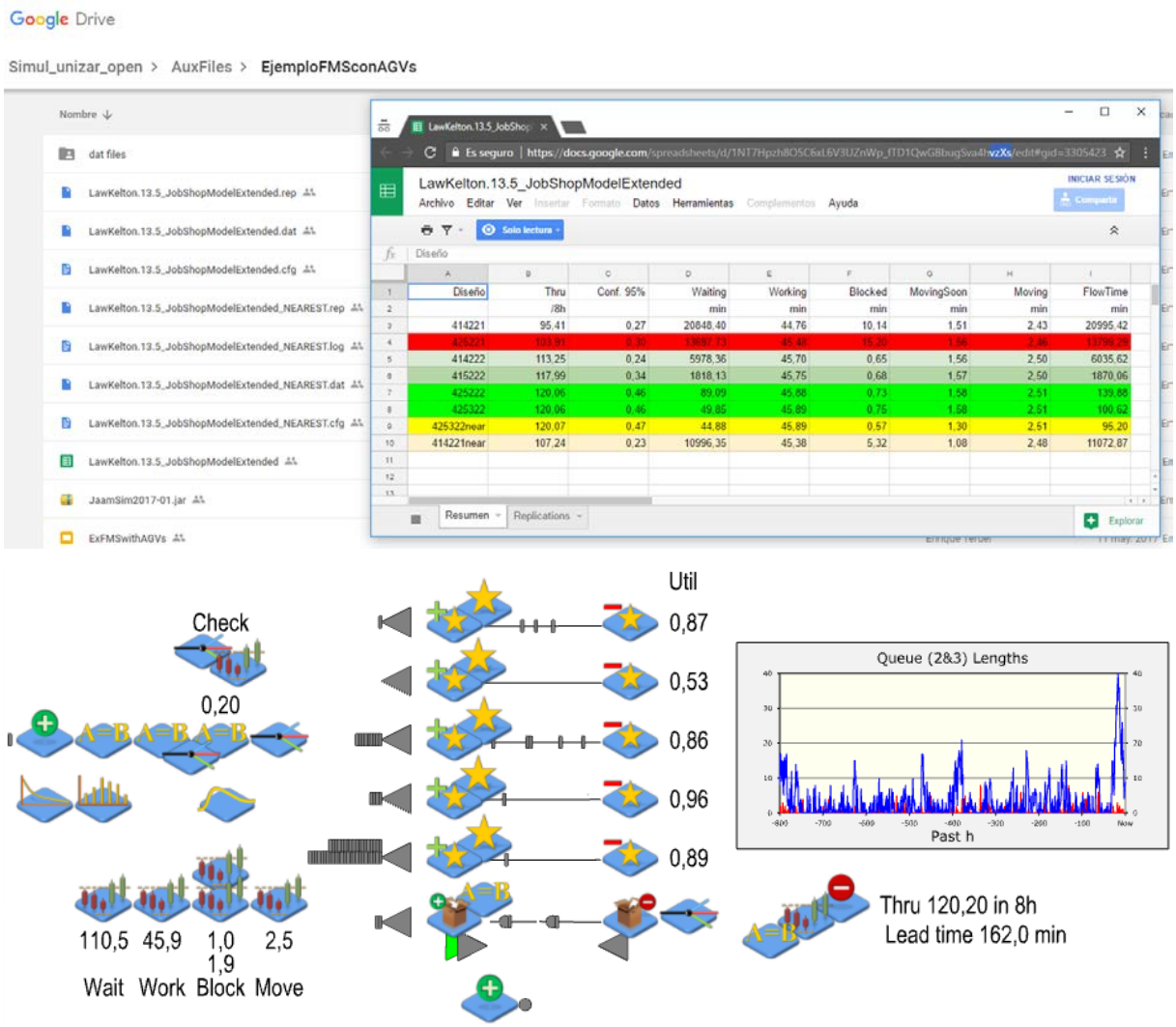


Figura 1: Ejemplo de Sistema de Fabricación Flexible con AGV's.
Carpeta, hoja de cálculo con resultados y modelo funcional en JaamSim.

en el modelado, facilidad de uso, rapidez y coste. En esta categoría JaamSim es más que satisfactoria, incluso aventaja a los paquetes comerciales más conocidos en rapidez de ejecución y tamaño de los sistemas que puede manejar, además de, obviamente, en coste. Permite simular múltiples réplicas de diversos escenarios utilizando un número arbitrario de índices en "Multiple Runs". También tiene la capacidad de modelar subsistemas continuos, útil para simular procesos híbridos como una bodega (proceso continuo hasta llegar al embotellado y empaquetado), aunque no hemos usado esta característica dado que hay otra parte de la asignatura centrada en ella. La facilidad de uso y flexibilidad de modelado son muy destacables, aunque hay margen de mejora, en particular convendrá dotar de capacidades para el modelado jerárquico, necesarias para grandes modelos con reutilización de submodelos (que ahora no puede hacerse en la interfaz gráfica, sí mediante programación), y también mejorar la usabilidad de la edición de expresiones (prevista a lo largo de los próximos meses, aunque en la situación actual se puede trabajar cómodamente con un poco de disciplina, editando fuera de JaamSim las expresiones y cortando-pegando).

- Requisitos informáticos. Son muy moderados, simplemente tener instalado Java. La instalación más conveniente, en cualquier entorno, es directamente descargar y ejecutar un "archivo Java" (.jar). No hemos tenido problemas con Windows (7 a 10) ni Ubuntu, aunque algún estudiante ha detectado fallos en MacOS. Ninguna otra herramienta, y no digamos las de código abierto más académicas, resulta tan fácil de poner en marcha, lo cual es una gran ventaja en general y más en el contexto en que nos movemos, en el que deseamos que los estudiantes trabajen con ella desde el primer día.
- Animación, y gráficos. En esta categoría las capacidades de JaamSim son sobresalientes. La animación, que puede incluir gráficos dinámicos, se puede acelerar con respecto a tiempo real, hacerse todo lo rápida que permita la máquina, u ocultarse para acelerarla aún más. Una característica muy útil, sobre todo en las etapas tempranas de modelado y verificación, es que muchos resultados estadísticos son directamente accesibles "en caliente", durante la animación, sin tener que esperar a que ésta termine. Además, es muy sencillo incorporar datos y gráficos de interés en la ventana de animación, como se ha ilustrado en el ejemplo.
- Características estadísticas, y salidas. Están disponibles las distribuciones más frecuentes, y bloques estadísticos para facilitar un análisis más que básico de las salidas, que por supuesto se puede/debe completar a partir de datos exportados, usando otras herramientas específicas. Una característica muy conveniente, para la comparación de alternativas, es la generación de números aleatorios comunes utilizando un único parámetro, "GlobalSubstreamSeed".
- Soporte y documentación. La documentación (para usuarios y programadores) es suficiente, aunque convendrá incorporar ejemplos de mayor complejidad en algún repositorio bien organizado, como los que nosotros podemos contribuir. El soporte es el típico de una comunidad, a través de un foro, en el que nuestra experiencia durante los últimos meses ha sido extraordinariamente positiva. Seguramente, si la comunidad crece más, habrá una atención menos directa por parte del equipo de desarrollo, pero más rica por parte del resto de usuarios.

Siguiendo las recomendaciones del modelo de calidad para evaluar software educativo libre (Touron *et al.*, 2015), valoraríamos JaamSim positivamente en todos las características que contempla dicho modelo, considerando que conviene seguir mejorando la confiabilidad y comunidad, ambas inter-relacionadas: durante nuestra experiencia como usuarios participando en la comunidad hemos podido detectar algún fallo así como proponer mejoras y nuevas funcionalidades, obteniendo siempre respuesta eficaz y a tiempo.

# 4 VALORACIÓN DE LA EXPERIENCIA, Y CONCLUSIONES

Consideramos que la experiencia ha cumplido con nuestros objetivos: los estudiantes han podido aprender simulación de eventos discretos trabajando directamente sobre modelos de diversa complejidad, los vistos en clase y otros desarrollados por ellos mismos, con un entorno de simulación fácil de instalar y usar, y a la vez suficientemente potente, sin ningún tipo de limitación. No era nuestro objetivo extraer conclusiones "objetivas", dado el reducido número de estudiantes en una sola edición del curso. Los resultados de las encuestas de calidad son muy positivos, en línea con los de otras asignaturas o los de anteriores ediciones de ésta, pero tampoco nos parecen significativos, por su carácter genérico y el reducido tamaño de la muestra. Para nuestra valoración u opinión positiva nos basamos en:

- La realimentación informal de los estudiantes, pues por desgracia ha sido imposible conseguir una realimentación más formal, mediante encuestas de satisfacción y valoración del esfuerzo requerido sobre las actividades específicas, que se les pidió que respondieran de forma voluntaria. Aunque en algunos casos

requería algo de trabajo extra, éste era mínimo, así que pensamos que la baja participación es más bien sistémica.

- La observación de su trabajo, cada vez más autónomo e interesado, especialmente desde el momento en que disponían del enunciado de su trabajo. Pronto dejaron de usar los ordenadores del laboratorio en las sesiones prácticas para usar los propios, cosa que también podían hacer en clase, y eso permitió disolver las fronteras entre las clases "de teoría" y "de prácticas", a menudo incómodas e inconvenientes. Esta observación nos sugiere adelantar el curso que viene la entrega de los enunciados de trabajo, de modo que casi desde el principio puedan y quieran ponerse a trabajar por su cuenta.
- La calidad de los trabajos prácticos realizados, cubriendo en general todas las fases de un proyecto de simulación, desde el planteamiento y definición de alcance y objetivos hasta el análisis estadístico de resultados y planteamiento de recomendaciones, pasando por el desarrollo de modelos y animaciones. Estos son todos los aspectos requeridos en un proyecto de simulación profesional, según (Altiok y Melamed, 2007; Law y Kelton, 2000). Varios grupos obtuvieron la máxima calificación, al haberlos cubierto todos excelentemente, a partir de un enunciado del tipo "blueprint" tomado directamente de los "Arena Contest Problems" de (Kelton *et al.*, 2015). En ediciones anteriores sólo se cubrían parcialmente, y sobre casos más simples y menos variados. Gracias a este tipo de evaluación, exigente e integradora, se ha considerado que esta asignatura contribuye a alcanzar la competencia transversal "Capacidad para combinar los conocimientos generalistas y los especializados de Ingeniería para generar propuestas innovadoras y competitivas en la actividad profesional", en un proyecto de "Planificación de las competencias transversales en los Grados de la Escuela de Ingeniería y Arquitectura" realizado este mismo curso.

## Agradecimientos

Agradecemos el apoyo recibido del Vicerrectorado de Política Académica de la Universidad de Zaragoza, a través del proyecto PIIDUZ-16-031.

## Referencias

Altiok, T. y Melamed, B. (2007). Simulation Modeling and Analysis with Arena. Academic Press.

Brown, P., Roedeiger III, H. y McDaniel, M. (2014). Make It Stick. The Science of Successful Learning. Harvard University Press.

Conole, G. (2013). Los MOOCs como tecnologías disruptivas: estrategias para mejorar la experiencia de aprendizaje y la calidad de los MOOCs. Campus Virtuales. Revista Científica Iberoamericana de Tecnología Educativa, 2(2), 16-28.

Dagkakis, G. y Heavey, C. (2016). A review of open source discrete event simulation software for operations research, Journal of Simulation (10), 193-206.

Fritzson, P. (2015). Principles of Object-Oriented Modeling and Simulation with Modelica 3.3: A Cyber-Physical Approach. Wiley.

JaamSim Development Team (2017). JaamSim: Discrete-Event Simulation Software (v.2017-03). JaamSim User Manual (v.2016-21). Abierto en línea en http://jaamsim.com/

Kelton, D., Sadowski y R., Zupick, N. (2015). Simulation with Arena. McGraw-Hill.

King, H. y Harrison, H. (2010). Discrete-event simulation in Java: a practitioner's experience, Proc. Conference on Grand Challenges in Modeling & Simulation, 436-441.

King, H. y Harrison, H. (2013). Open-source simulation software: JaamSim, Proc. 2013 Winter Simulation Conference, 2163-2171.

Law, A. y Kelton, D. (2000). Simulation Modeling and Analysis. McGraw-Hill.

Leemis, L. y Park, S. (2006). Discrete-Event Simulation: A First Course. Pearson.

Martín, C., Urquía, A. y Dormido, S. (2005). Modeling of Interactive Virtual Laboratories with Modelica, Proceedings of the 4th International Modelica Conference, 159-168.

Mattsson, S., Elmqvist, H. y Otter, M. (1998). Physical system modeling with Modelica. Control Engineering Practice (6), 501-510.

Teruel, E. (2017). Simulación de Sistemas Dinámicos. Abierto en línea, a través de Internet Archive https://archive.org/details/Teruel2017SimulacionSistemasDinamicos

Tiller, M. (2014). Modelica by example. Abierto en línea en http://book.xogeny.com

Tourón, M., Plaza, I., Igual, R., Sainz, E. y Arcega, F. (2015). Modelo de calidad para evaluar software educativo libre, Proc. III Congreso Internacional sobre Aprendizaje, Innovación y Competitividad, 585-590.