Guía docente de la asignatura

Asignatura	DESARROLLO DE APLICACIONES DISTRIBUIDAS		
Materia	INGENIERÍA DE REDES, SISTEMAS Y SERVICIOS TELEMÁTICOS		
Módulo	MATERIAS ESPECÍFICAS DE TECNOLOGÍAS DE TELECOMUNICACIÓN		
Titulación	GRADO EN INGENIERÍA DE TECNOLOGÍAS DE TELECOMUNICACIÓN		
Plan	460	**Código**	45020
Periodo de impartición	1er CUATRIMESTRE	**Tipo/Carácter**	OBLIGATORIA
Nivel/Ciclo	GRADO	**Curso**	3º
Créditos ECTS	6 ECTS		
Lengua en que se imparte	CASTELLANO		
Profesor/es responsable/s	NOMBRE DEL PROFESOR 1 (de momento en blanco) NOMBRE DEL PROFESOR 2 (de momento en blanco)		
Datos de contacto (E-mail, teléfono…)	TELÉFONO: 983 423000 ext. 1234 / ext. 5678 E-MAIL: profesor1@dpto.uva.es, profesor2@dpto.uva.es		
Horario de tutorías	Véase www.uva.es → Centros → Campus de Valladolid → Escuela Técnica Superior de Ingenieros de Telecomunicación → Tutorías		
Departamento	TEORÍA DE LA SEÑAL Y COMUNICACIONES E INGENIERÍA TELEMÁTICA		

1. Situación / Sentido de la Asignatura

1.1 Contextualización

La aparición y generalización de las redes de ordenadores ha dado lugar a la aparición de dos nuevos tipos de aplicaciones: las aplicaciones en red y las aplicaciones distribuidas. Estas aplicaciones se caracterizan por estar constituidas por varias partes que se pueden ejecutar en diferentes ordenadores. La coordinación entre estas partes se lleva a cabo mediante el intercambio de información a través de la red.

El desarrollo de una aplicación en red supone un esfuerzo superior al de una aplicación centralizada, ya que el desarrollador debe considerar todas las tareas de bajo nivel necesarias para el envío y recepción de información a través de la red y la gestión de los errores que puedan surgir durante esta comunicación. El desarrollo de las aplicaciones en red se basa en el uso de una API (interfaz de programación de aplicaciones) de acceso a los servicios de transporte de datos proporcionados por el sistema operativo. Esta API permite la solicitud de operaciones de bajo nivel relacionadas con el uso de la red (establecimiento y liberación de conexiones, envío y recepción de paquetes, etc.)

Para simplificar el desarrollo de las aplicaciones en red, de forma que la complejidad sea similar a la del desarrollo de una aplicación centralizada, surge el concepto de *middleware* o software intermediario. El *middleware* es, básicamente, una capa de software situada entre el sistema operativo y la aplicación a desarrollar, y que permite a la aplicación usar funciones de alto nivel (ejecución de una función en un ordenador remoto) sin necesidad de usar explícitamente las funciones de bajo nivel necesarias para la interacción con la red de comunicación. ~~De esta forma, la solicitud de la ejecución de código en un ordenador remoto se realiza de manera muy similar a la solicitud de ejecución de una función local.~~ La aparición del *middleware* ha hecho surgir un nuevo tipo de aplicaciones, las aplicaciones distribuidas, que básicamente se diferencian de las aplicaciones en red en que el desarrollador no necesita incluir código para acceder a las funciones de bajo nivel relacionadas con la red (las operaciones sobre la red necesarias para acceder a un recurso remoto se realizan implícitamente mediante el software intermediario ~~*middleware*~~). En este concepto de *middleware* se basan las tecnologías para el desarrollo de aplicaciones distribuidas utilizadas mayoritariamente en la actualidad, como pueden ser *enterprise java beans, .Net o servicios web.*

~~En este concepto de *middleware* se basan las tecnologías para el desarrollo de aplicaciones distribuidas utilizadas mayoritariamente en la actualidad, como pueden ser *enterprise java beans, .Net o servicios web.*~~

La situación actual con redes ubicuas y de relativamente altas prestaciones ha dado lugar a un gran número ~~todo tipo~~ de aplicaciones que necesitan usar una red de comunicaciones para proporcionar las funciones para las que han sido diseñadas. Por este motivo, resulta necesario en la actualidad formar profesionales que comprendan los principales problemas inherentes al desarrollo de aplicaciones en red y distribuidas y conozcan las principales tecnologías disponibles para desarrollar este tipo de aplicaciones de forma eficiente.

En esta asignatura se analizarán las principales diferencias entre aplicaciones centralizadas, en red y distribuidas, se describirán los principales tipos de *middleware* existentes (orientado a objetos, orientado a

componentes y orientado a servicios) y se explicará e ilustrará con ejemplos el proceso de desarrollo de aplicaciones en red y distribuidas siguiendo distintas tecnologías ampliamente usadas en la actualidad y que están basadas en los distintos tipos de *middleware* previamente presentados.

1.2 Relación con otras materias

Esta asignatura se apoya en la asignatura "Programación" de la materia "Informática" del "Bloque de Materias Instrumentales" que se imparte en el primer cuatrimestre del primer curso y en la asignatura de “Ingeniería de Sistemas Software” de la materia “Fundamentos de Sistemas Software” del “Bloque de Materias Básicas de Telecomunicaciones” que se imparte en el segundo cuatrimestre del segundo curso. En dichas asignaturas se proporcionarán los conceptos básicos de programación y desarrollo de sistemas software que facilitarán al alumno la comprensión de otros mostrados en la presente asignatura.

Además, también se utilizarán en esta asignatura conceptos relativos a la capa de transporte y red de la arquitectura de protocolos TCP/IP que han sido introducidos en la asignatura “Arquitectura de Redes, Sistemas y Servicios” de la materia “Fundamentos de Protocolos, Redes y Servicios Telemáticos” del “Bloque de Materias Básicas de Telecomunicaciones”, y que se imparte en el primer cuatrimestre del segundo curso.

1.3 Prerrequisitos

No existen condiciones previas excluyentes para cursar esta asignatura, aunque sí recomendaciones lógicas que el alumno debería tener en cuenta. En concreto, es recomendable haber cursado con anterioridad las asignaturas de “Programación”, “Ingeniería de Sistemas Software”, y “ Arquitectura de Redes, Sistemas y Servicios”.

2. Competencias

2.1 Generales

- GBE3. Capacidad para resolver problemas con iniciativa, creatividad y razonamiento crítico.
- GC1. Capacidad de organización, planificación y gestión del tiempo.
- GC2. Capacidad para comunicar, tanto por escrito como de forma oral, conocimientos, procedimientos, resultados e ideas relacionadas con las telecomunicaciones y la electrónica.
- GC3. Trabajar en cualquier contexto, individual o en grupo, de aprendizaje o profesional, local o internacional, desde el respeto a los derechos fundamentales, de igualdad de sexo, raza o religión y los principios de accesibilidad universal, así como la cultura de paz.

2.2 Específicas

- T7 Conocimiento y utilización de los fundamentos de la programación en redes, sistemas y servicios de telecomunicación
- T13. Capacidad de diferenciar los conceptos de redes de acceso y transporte, redes de conmutación de circuitos y de paquetes, redes fijas y móviles, así como los sistemas y aplicaciones de red distribuidos, servicios de voz, datos, audio, vídeo y servicios interactivos y multimedia.
- TEL6. Capacidad de diseñar arquitecturas de redes y servicios telemáticos.
- TEL7. Capacidad de programación de servicios y aplicaciones telemáticas, en red y distribuidas.

3. Objetivos

Al finalizar la asignatura el alumno deberá ser capaz de:

- Comprender la problemática específica del desarrollo de aplicaciones en red.
- Comprender los conceptos relacionados con el *middleware* como arquitectura básica para el desarrollo de aplicaciones distribuidas.
- ~~Conocer~~Describir y comparar los distintos tipos de *middleware*.
- ~~Conocer~~Analizar las principales soluciones existentes en el ámbito del *middleware* orientado a objetos, a componentes y a servicios .
- Diseñar, desarrollar y desplegar aplicaciones en red utilizando una API de acceso a los servicios de transporte de datos.
- Diseñar, desarrollar y desplegar aplicaciones distribuidas utilizando distintas tecnologías basadas en *middleware*.

4. Tabla de dedicación del estudiante a la asignatura

ACTIVIDADES PRESENCIALES	HORAS	ACTIVIDADES NO PRESENCIALES	HORAS
Clases teórico-prácticas (T/M)	30	Estudio y trabajo autónomo individual	45
Clases prácticas de aula (A)	0	Estudio y trabajo autónomo grupal	45
Laboratorios (L)	30		
Prácticas externas, clínicas o de campo	0		
Seminarios (S)	0		
Tutorías grupales (TG)			
Evaluación (fuera del periodo oficial de exámenes)	0		
Total presencial	**60**	**Total no presencial**	**90**

5. Bloques temáticos

Bloque 1: Desarrollo de aplicaciones distribuidas

Carga de trabajo en créditos ECTS: 6

a. Contextualización y justificación

Véase el apartado 1.1.

b. Objetivos de aprendizaje

Véase el apartado 3.

c. Contenidos

TEMA 0: Introducción al lenguaje java

- 0.1 El paradigma de orientación a objetos
- 0.2 Principales construcciones del lenguaje
- 0.3 Ejemplo de desarrollo de aplicación en el lenguaje java

TEMA 1: Introducción a los sistemas distribuidos y el middleware

- 1.1 Conceptos básicos: sistemas centralizados, redes de ordenadores y sistemas distribuidos
- 1.2 Requisitos para los sistemas distribuidos
- 1.3 Tipos de transparencia en los sistemas distribuidos
- 1.4 Tipos de software distribuido:
  - 1.4.1. Aplicaciones en red
  - 1.4.2. Software para sistemas operativos distribuidos
  - *1.4.3.* Aplicaciones basadas en *middleware*
- 1.5 Ejemplo de desarrollo de una aplicación en red basadas en APIs de acceso a los servicios de transporte de datos
- 1.6 Resumen

TEMA 2: Middleware orientado a objetos

- 2.1 Características comunes
- 2.2 Sun: Java RMI
- 2.3 OMG: CORBA
- 2.4 Microsoft: COM/DCOM
- 2.5 Ejemplo de desarrollo de una aplicación distribuida
- 2.6 Resumen

TEMA 3: Middleware orientado a componentes

3.1 Características comunes
3.2 Sun: J2EE / EJB
3.3 OMG: CORBA/CCM
3.4 Microsoft: .Net
3.5 Ejemplo de desarrollo de una aplicación distribuida
3.6 Resumen

TEMA 4: Middleware orientado a servicios

4.1 Características comunes
4.2 Servicios web
4.3 Ejemplo de desarrollo de una aplicación distribuida
4.4 Resumen

d. Métodos docentes

- Clase magistral participativa
- Taller de prácticas guiadas en el laboratorio

e. Plan de trabajo

Véase el Anexo I.

f. Evaluación

La evaluación de la adquisición de competencias se basará en:

- Prueba escrita al final del cuatrimestre
- Funcionamiento y estructura de los programas desarrollados en el laboratorio por el alumno, así como la documentación asociada a dichos programas
- Respuestas a las preguntas planteadas en los enunciados de prácticas

g. Bibliografía básica

- G. F. Coulouris, J. Dollimore, y T. Kindberg, *Distributed systems: concepts and design*, 5.ª ed. Addison Wesley. Madrid, 2012.
- R. W. Stevens, *Unix Network Programming. Networking APIs: sockets and XTI*, 2.ª ed. Prentice-Hall, 1998.
- W. Emmerich, *Engineering distributed objects*. John Wiley & Sons, 2001.
- ~~J. Pritchard, *COM and CORBA side by side: architectures, strategies, and implementations*. Addison Wesley, 1999.~~
- C. Szyperski, *Component software: beyond object oriented programming*. Addison Wesley, 1999.
- ~~Oracle: *The Java EE 6 tutorial*. March 2011.~~
- ~~Object Managment Group: Tutorial sobre CCM.~~

~~http://www.omg.org/news/meetings/workshops/RT_2003_Manual/Tutorials/T3_CCM_Wang-Rodrigues.pdf~~

- ~~Microsoft: *.NET Framework Conceptual Overview*.~~
- ~~http://msdn.microsoft.com/en-us/library/zw4w595w.aspx.~~

- G. Alonso, F. Casati, H. Kuno, y V. Machiraju, *Web Services: Concepts, Architectures and Applications*. Springer, 2004.
- L. Richardson, S. Ruby, y D. H. Hansson, *RESTful web services*. O'Reilly, 2007.

h. Bibliografía complementaria

- J. Pritchard, *COM and CORBA side by side: architectures, strategies, and implementations*. Addison-Wesley, 1999.
- G. Suresh Raj, *A Detailed Comparison of CORBA, DCOM and Java/RMI*. 2000. http://my.execpc.com/~gopalan/misc/compare.html
- Oracle: *Java Platform, Standard Edition 7 API Specification*. 2011. http://java.sun.com/javase/7/docs/api/
- Oracle: *Java Platform, Enterprise Edition (Java EE). Documentation home page.* http://java.sun.com/j2ee
- Oracle: *The java tutorials: RMI*. 2012. http://download.oracle.com/javase/tutorial/rmi/index.html
- Oracle: *Dynamic code downloading using RMI*. 2003. http://download.oracle.com/javase/1.4.2/docs/guide/rmi/codebase.html
- Oracle: *Java SE documentation: Java Remote Method Invocation (Java RMI)*. 2011. http://download.oracle.com/javase/7/docs/technotes/guides/rmi/index.html
- Oracle: *Java SE documentation: Java RMI over IIOP*. 2011. http://download.oracle.com/javase/7/docs/technotes/guides/rmi-iiop/index.html
- *The java tutorials: Java Naming and Directory Interface*. 2012. http://download.oracle.com/javase/tutorial/jndi/index.html
- Oracle: *Java SE Documentation. Java IDL (CORBA).* 2011. http://java.sun.com/javase/6/docs/technotes/guides/idl/
- *Microsoft Component Object Model Technologies.* 2012. http://www.microsoft.com/com/default.mspx
- E. Roman, S. Ambler, y T. Jewell, *Mastering Enterprise JavaBeans*, 2.ª ed. Wiley, 2002.
- D. Sevilla, *Tutorial de CCM*. http://ditec.um.es/~dsevilla/ccm/#CCMtutorial
- D. C. Schmidt, *Página web de Douglas C. Schmidt sobre CCM*. http://www.cs.wustl.edu/~schmidt/corba.html
- G. Suresh Raj, *Component Engineering Cornucopia*.. http://my.execpc.com/~gopalan/
- Object Managment Group: Tutorial sobre CCM. http://www.omg.org/news/meetings/workshops/RT_2003_Manual/Tutorials/T3_CCM_Wang-Rodrigues.pdf
- Microsoft: *Microsoft .NET*. http://www.microsoft.com/net
- Codeguru: *The .NET Architecture*. http://www.codeguru.com/csharp/sample_chapter/article.php/c8245
- W3C: *Web of services. 2012.* http://www.w3.org/standards/webofservices/
- W3C: *Web Services Activity*. 2011. http://www.w3.org/2002/ws

i. Recursos necesarios

- Documentación de apoyo
- Entorno de trabajo en la plataforma Moodle ubicada en el Campus Virtual de la Universidad de Valladolid u otra plataforma virtual alternativa.

- Laboratorio de prácticas, con al menos un ordenador para cada dos alumnos, para las sesiones de laboratorio. Cada ordenador contará con el kit de desarrollo de java SE y EE, y los entornos de desarrollo bluej y netbeans IDE.

6. Temporalización (por bloques temáticos)

BLOQUE TEMÁTICO	CARGA ECTS	PERIODO PREVISTO DE DESARROLLO
Bloque 1: Desarrollo de aplicaciones distribuidas	6 ECTS	Semanas 1 a 15

7. Sistema de calificaciones – Tabla resumen

INSTRUMENTO/PROCEDIMIENTO	PESO EN LA NOTA FINAL	OBSERVACIONES
Informes de prácticas de laboratorio y revisión del funcionamiento de los programas desarrollados	50%	Es condición necesaria (pero no suficiente) para superar la asignatura alcanzar una calificación igual o superior a 5 sobre 10 puntos en la nota total de laboratorio (informes+funcionamiento de los programas). En cada práctica, la nota total será el promedio de la nota sobre el funcionamiento del programa y de la nota del informe asociado.
Examen final escrito	50%	Es condición necesaria (pero no suficiente) para superar la asignatura alcanzar una calificación igual o superior a 5 sobre 10 puntos en este examen.

En el caso de la convocatoria extraordinaria:

- Se mantiene (dentro de un mismo curso académico) la calificación obtenida en cada instrumento de la tabla siempre que dicha calificación sea igual o superior a 5 sobre 10 puntos
- El alumno deberá realizar de nuevo el examen final escrito si la nota obtenida en él para convocatorias anteriores del mismo curso académico es inferior a 5 sobre 10 puntos
- El alumno deberá realizar de nuevo la práctica o prácticas suspensas (puntuación inferior a 5 sobre 10 puntos) si la nota total de prácticas obtenida en convocatorias anteriores del mismo curso académico es inferior a 5 sobre 10 puntos

8. Consideraciones finales

El Anexo I mencionado en la guía, donde se describe la planificación detallada, se entregará al comienzo de la asignatura.