PROGRAMA DE LA ASIGNATURA

Curso académico: 2017-2018

Identificación y características de la asignatura					
Código	501284			Créditos ECTS	6
Denominación (español)	Programación concurrente y distribuida				
Denominación (inglés)	Concurrent and Distributed Programming				
Titulaciones	Grado en Ingeniería Informática en Ingeniería de Computadores Grado en Ingeniería Informática en Ingeniería del Software				
Centro	Escuela Politécnica				
Semestre	Cuarto	Carácter	Obligatorio		
Módulo	Común a la rama de Informática				
Materia	Programación				

Profesor/es			
Nombre	Despacho	Correo-e	Página web
Juan Hernández Núñez	Edificio Investigación Lab. Ingeniería del Software	juanher@unex.es	http://www.unex.es/investigacion/grupos/quercus
Fernando Sánchez Figueroa	08. Pabellón de Informática	fernando@unex.es	http://www.unex.es/investigacion/grupos/quercus

Área de conocimiento	Lenguajes y Sistemas Informáticos
Departamento	Ingeniería de Sistemas Informáticos y Telemáticos
Profesor coordinador (si hay más de uno)	Fernando Sánchez Figueroa

Competencias

Competencias básicas

(Competencias básicas establecidas para Grado en el Anexo I 3.2 del RD 861/2010)

CB1: Que los estudiantes hayan demostrado poseer y comprender conocimientos en un área de estudio que parte de la base de la educación secundaria general, y se suele encontrar a un nivel que, si bien se apoya en libros de texto avanzados, incluye también algunos aspectos que implican conocimientos procedentes de la vanguardia de su campo de estudio.
CB2: Que los estudiantes sepan aplicar sus conocimientos a su trabajo o vocación de una forma profesional y posean las competencias que suelen demostrarse por medio de la elaboración y defensa de argumentos y la resolución de problemas dentro de su área de estudio.
CB3: Que los estudiantes tengan la capacidad de reunir e interpretar datos relevantes (normalmente dentro de su área de estudio) para emitir juicios que incluyan una reflexión sobre temas relevantes de índole social, científica o ética.
CB4: Que los estudiantes puedan transmitir información, ideas, problemas y soluciones a un público tanto especializado como no especializado.
CB5: Que los estudiantes hayan desarrollado aquellas habilidades de aprendizaje necesarias para emprender estudios posteriores con un alto grado de autonomía.

Competencias específicas

Según los planes de estudio aprobados, esta asignatura debe cubrir, total o parcialmente, las siguientes competencias específicas y sus resultados de aprendizaje.

CI11: Conocimiento y aplicación de las características, funcionalidades y estructura de los Sistemas Distribuidos, las Redes de Computadores e Internet y diseñar e implementar aplicaciones basadas en ellas.
CI14: Conocimiento y aplicación de los principios fundamentales y técnicas básicas de la programación paralela, concurrente, distribuida y de tiempo real

Competencias transversales

Según los planes de estudio aprobados y los acuerdos de la comisión de calidad de las titulaciones, esta asignatura debe cubrir, total o parcialmente, las siguientes competencias transversales y sus resultados de aprendizaje en un nivel intermedio.

CT09: Capacidad de trabajo en grupo
CT16: Capacidad para adaptarse a nuevas situaciones y cambios

La competencia transversal CT09 y sus resultados de aprendizaje se complementan en la asignatura "*Inteligencia Artificial y Sistemas Inteligentes*", también del 4º semestre.

Contenidos

Breve descripción del contenido

Concurrencia y distribución. Primitivas de sincronización. Desarrollo de aplicaciones concurrentes y distribuidas.

Temario de la asignatura

TEMA 1 CONCEPTOS FUNDAMENTALES DE LA PROGRAMACIÓN CONCURRENTE Y DISTRIBUIDA

- 1.1 Introducción
- 1.2 Concepto de programación concurrente
- 1.3 Beneficios de la programación concurrente
- 1.4 Concurrencia y arquitecturas hardware
- 1.5 Especificación de ejecución concurrente
- 1.6 Características de los sistemas concurrentes
- 1.7 Problemas inherentes a la programación concurrente
- 1.8 Corrección de programas concurrentes
- 1.9 Concepto de programación distribuida

TEMA 2 PROBLEMAS DE SINCRONIZACIÓN. PRIMERAS SOLUCIONES

- 2.1 Tipos de sincronización
- 2.2 Solución a la condición de sincronización
- 2.3 Soluciones a la exclusión mutua
  - 2.3.1 Soluciones Software
  - 2.3.2 Soluciones Hardware
  - 2.3.3 Otras soluciones: deshabilitación de interrupciones

TEMA 3 PROGRAMACIÓN MULTIHILO EN JAVA

- 3.1 Procesos vs. Hilos
- 3.2 Creación de threads en Java
- 3.3 Sincronización de threads en Java
- 3.4 Resolución de problemas

TEMA 4 PRIMITIVAS DE SINCRONIZACIÓN EN MEMORIA COMPARTIDA

- 4.1 Semáforos
  - 4.1.1. Introducción
  - 4.1.2. Definición de semáforo
  - 4.1.3. Resolución de problemas usando semáforos
  - 4.1.4. Implementación de semáforos
  - 4.1.5. Inconvenientes de semáforos
- 4.2 Monitores
  - 4.2.1 Introducción
  - 4.2.2 Definición de monitor
  - 4.2.3 Condición de sincronización en monitores
  - 4.2.4 Resolución de problemas usando monitores
  - 4.2.5 Equivalencia entre monitores y semáforos
  - 4.2.6 Semántica de la operación signal
  - 4.2.7 Llamadas anidadas de monitores

TEMA 5 PROGRAMACIÓN CONCURRENTE AVANZADA EN JAVA

- 5.1 Estructuras de datos concurrentes
- 5.2 Creación avanzada de threads
- 5.3 Mecanismos de sincronización avanzados

TEMA 6 INTERBLOQUEOS

- 6.1 Definición del interbloqueo
- 6.2 Caracterización del interbloqueo
- 6.3 Tratamiento del interbloqueo

TEMA 7 MECANISMOS DE PASO DE MENSAJE

- 7.1 Introducción
- 7.2 Identificación en el proceso de comunicación
- 7.3 Paso de mensaje síncrono
- 7.4 Paso de mensaje asíncrono
- 7.5 Invocación remota
- 7.6 Características del medio de transmisión
- 7.7 Espera Selectiva
- 7.8 Paso de mensajes en Java
  - 7.8.1 Sockets TCP
  - 7.8.2 Sockets UDP

TEMA 8. PROBLEMAS DE LA PROGRAMACIÓN DISTRIBUIDA.

- 8.1. Introducción.
- 8.2. Técnicas básicas de programación distribuida.
- 8.3. Problemas básicos en la programación distribuida.
  - 8.3.1 La exclusión mutua distribuida.
  - 8.3.2 La Detección de terminación distribuida.
  - 8.3.3 La detección del deadlock en la programación distribuida

TEMA 9. MODELOS Y LENGUAJES BASADOS EN COMPARTICIÓN DE MEMORIA.

- 9.1 Introducción.
- 9.2 Espacios de tuplas.
- 9.3 El modelo de Linda.
- 9.4 Implementación del modelo de Linda en Java

TEMA 10. PLATAFORMAS DE COMPONENTES DISTRIBUIDOS.

- 10.1 Plataformas de componentes: Conceptos y principios.
- 10.2 Sistemas de Objetos Distribuidos
- 10.3 Computación distribuida con Java RMI.
- 10.4 Migración de objetos distribuidos
- 10.5 Callbacks

Actividades formativas					
Horas de trabajo del alumno por tema		**Presencial**		**Actividad de seguimiento**	**No presencial**
Tema	**Total**	**GG**	**SL**	**TP**	**EP**
1	4,5	1,5	0,5	0	2,5
2	8	2	1	0	5
3	22	5	3	01	14
4	18	4	3	0	11
5	15	2,5	2,5	0	10
6	9	2	0	1	6
7	14	2	2	0	10,5
8	11	3	1,5	0	6
9	6	2	0	0	4
10	38	7	7	1	23
Evaluación del conjunto	4,5	3	1,5	0	0
TOTAL	**150**	**34**	**22**	**2**	**92**

Temas relativos a la programación concurrente
Temas relativos a la programación distribuida

GG: Grupo Grande (100 estudiantes).
SL: Seminario/Laboratorio (prácticas clínicas hospitalarias = 7 estudiantes; prácticas laboratorio o campo = 15; prácticas sala ordenador o laboratorio de idiomas = 30, clases problemas o seminarios o casos prácticos = 40).
TP: Tutorías Programadas (seguimiento docente, tipo tutorías ECTS).
EP: Estudio personal, trabajos individuales o en grupo, y lectura de bibliografía.

Metodologías docentes*

La asignatura "Programación concurrente y distribuida" busca la participación activa y continuada de los estudiantes, quienes deberán hacer frente a nuevos retos que se irán proponiendo a lo largo de la asignatura, y donde se hará un uso intensivo del Campus Virtual.
A continuación, se detallan algunas de las actividades formativas que se plantearán a lo largo del curso para alcanzar los objetivos de aprendizaje de la asignatura. Aunque cada actividad sólo se detalla dentro de una modalidad (presenciales en grupo grande, presenciales en laboratorio, tutorías ECTS y no presenciales), algunas de ellas se desarrollarán en varias pudiendo, éstas, ser realizadas de forma individual y/o en grupo.

Presenciales en grupo grande

Orientadas principalmente a la adquisición de los conceptos teóricos de la asignatura, en estas actividades se combinan las clases expositivas con la resolución de problemas individualmente y/o en grupo con metodologías activas de aprendizaje.

Presenciales en laboratorio

Las sesiones de laboratorio estarán a disposición de los alumnos antes del inicio de cada sesión. Cada sesión dispone de un guión que contiene los objetivos y los trabajos que se deben desarrollar. El estudiante deberá realizar parte de alguna de las sesiones previamente y de manera autónoma, de manera que en el momento de la sesión presencial de laboratorio sea capaz de implantar la solución software real. En ese caso, se seguirá una metodología de Flipped ClassRoom.

Los grupos de laboratorio tendrán un número máximo de 12 alumnos.

Tutorías ECTS

Las actividades formativas que se plantean en este bloque están orientadas, principalmente, a realizar el seguimiento de la adquisición de las competencias transversales.
Mediante una estrategia de roles, se plantearán diversos problemas que el grupo debe ir resolviendo a lo largo del curso. Los requisitos de cada uno de estos problemas son cambiantes, de manera que el grupo debe hacer frente a las nuevas situaciones y cambios que requieren los nuevos requisitos. Se hará uso de rúbricas para determinar el grado de consecución de las actividades propuestas.

Cada grupo de laboratorio se dividirá en 2 o 3 grupos ECTS cada uno.

No Presenciales

Dentro de las actividades no presenciales planteadas se encuentran las siguientes:

- Visualización de videos y actividades preparatorias de las clases, al estilo "flipped classroom"
- Estudio individual.
- Reuniones de grupo
- Búsqueda de información
- Plantear preguntas de test
- Seguimiento de problemas resueltos
- Acceso a documentación del aula virtual
- Comunicación con profesores y compañeros mediante foros
- Cuestionarios de evaluación y autoevaluación

Resultados de aprendizaje*

Asociados a las competencias básicas y específicas:

- Puede utilizar de manera eficaz un entorno de programación que incluya herramientas de edición, compilación, depuración y documentación de programas.
- Justifica la utilización de distintos paradigmas de programación y plataformas de desarrollo de software en un determinado contexto.
- Busca, analiza, sintetiza y critica nueva información para aprender nuevos lenguajes, algoritmos, técnicas, paradigmas y metodologías de programación aplicables a distintas áreas, teniendo como objetivo la actualización continua de los conocimientos y competencias.
- Conoce las principales primitivas relacionadas con la concurrencia y las aplica en el diseño de este tipo de sistemas.
- Conoce los conceptos fundamentales sobre sistemas de computación distribuida y sus distintas aplicaciones.

Asociados a las competencias transversales:

CT09:

- Conoce las normas básicas de trabajo en equipo, colaboración, compromiso y responsabilidad y las técnicas básicas de trabajo
- Conoce y aplica técnicas básicas de trabajo en equipos que trabajan de forma presencial o virtual.
- Trabaja de manera eficiente como parte integrante o liderando equipos unidisciplinares o multidisciplinares.
- Contribuye al trabajo del grupo y favorece la buena comunicación, pudiendo desempeñar distintas funciones dentro del grupo.
- Participa en el establecimiento de planes de trabajo equilibrados y efectivos, y evalúa su ejecución.

CT16:

- Identifica las situaciones de cambio.
- Elabora las estrategias para abordar la problemática implicada por la nueva situación, aceptando ser flexible y estando dispuesto a cambiar las propias ideas ante una nueva información o vivencia contraria.

Objetivos de aprendizaje

Para desarrollar adecuadamente las competencias asignadas a esta asignatura y poder alcanzar los resultados de aprendizaje propuestos, se establecen los siguientes objetivos de aprendizaje específicos:

Relacionados con competencias académicas y disciplinares:

- Dotar al alumno de un conocimiento general sobre la Programación concurrente y distribuida.
- Entender y distinguir cuándo un problema es de naturaleza concurrente y/o distribuida.
- Conocer e identificar los distintos problemas que se plantean en la programación de aplicaciones para sistemas concurrentes y distribuidos, aplicando soluciones efectivas a los mismos.
- Conocer las principales primitivas de sincronización tanto en memoria compartida como distribuida.
- Dado un problema de naturaleza concurrente, decidir qué tipo de primitiva es más apropiada
- Resolver problemas de naturaleza concurrente y distribuida
- Conocer y aplicar de forma efectiva los distintos mecanismos de paso de mensaje.
- Conocer distintos modelos de lenguajes para la programación distribuida, entre los que destacan dos grandes familias: modelos basados en compartición de memoria y modelos basados en paso de mensajes.
- Conocer los principios que rigen a la computación distribuida orientada a objetos, con especial énfasis en Java/RMI.

Relacionados con otras competencias personales y profesionales:

- Planificar y organizar el trabajo personal.
- Tener iniciativa y ser resolutivo, aportando soluciones efectivas a los problemas planteados incluso en situaciones de falta de información y/o con restricciones temporales y/o de recursos.
- Mostrar una adecuada capacidad de relación interpersonal
- Encontrar, analizar, criticar, relacionar, estructurar y sintetizar información proveniente de diversas fuentes, así como integrar ideas y conocimientos.
- Ser capaz de argumentar y justificar lógicamente las decisiones, sabiendo aceptar otros puntos de vista.
- Expresar expectativas positivas del equipo de trabajo, hablar bien de los demás miembros del grupo, demostrar respeto, mantener una actitud abierta a aprender de los otros miembros del grupo.
- Solicitar opiniones e ideas a la hora de tomar decisiones específicas o hacer planes, valorando las ideas y experiencias de los demás, atendiendo, escuchando y promoviendo la cooperación de los miembros del grupo

Sistemas de evaluación

Se aplicarán el sistema de calificaciones vigente en el RD 1125/2003, artículo 5º y la RESOLUCIÓN de 25 de noviembre de 2016, de la Gerencia e la Universidad de Extremadura, por la que se ejecuta el Acuerdo adoptado por el Consejo de Gobierno por el que se aprueba la modificación de la normativa de evaluación de los resultados de aprendizaje y de las competencias adquiridas por el alumnado en las titulaciones oficiales de la Universidad de Extremadura (DOE del 12-12-2016)

Para la evaluación de la asignatura existirán dos modalidades diferentes: evaluación continua y evaluación final. La elección entre el sistema de evaluación continua o el sistema de evaluación por prueba final de carácter global corresponde al estudiante durante las tres primeras semanas de cada semestre (Art. 4.6 de la Normativa de Evaluación de la UEx (DOE 12/12/2016)).

Modalidad Evaluación Continua.

Criterios de evaluación

Para aprobar la asignatura el estudiante deberá:

- Superar la evaluación de los conocimientos teóricos con los requisitos mínimos propuestos en cada uno de los bloques que se detallan a continuación.
- Superar la evaluación de los conocimientos prácticos con los requisitos mínimos propuestos en cada uno de los bloques que se detallan a continuación.
- Demostrar la adquisición de las competencias transversales (CT09 y CT16) mediante la realización de un proyecto de programación en grupo referido en el bloque 3 (evaluación continua)
- La puntuación de cada bloque se calculará sobre 10.

Bloque 1: Concurrente (temas 1-6)

- Este bloque es recuperable y su calificación será del 40% de la calificación total en la asignatura.
- Es obligatorio superar este bloque con una nota mínima de 3,5 para aprobar la asignatura.
- La evaluación de este bloque será mediante:
    - un examen escrito de contenidos teóricos,
    - una prueba práctica que podrá ser escrita o en ordenador.
- Es obligatorio superar cada uno de los exámenes teóricos y prácticos con una nota mínima de 3,5 para aprobar este bloque.

Bloque 2: Distribuida (temas 7-10)

- Este bloque es recuperable y su calificación será del 40% de la calificación total en la asignatura.
- Es obligatorio superar este bloque con una nota mínima de 3,5 para aprobar la asignatura.
- La evaluación de este bloque será mediante:
    - un examen escrito de contenidos teóricos
    - una prueba práctica que podrá ser escrita o en ordenador.
- Es obligatorio superar cada uno de los exámenes teóricos y prácticos con una nota mínima de 3,5 para aprobar este bloque.
-

Bloque 3: Evaluación continua (actividades de autoaprendizaje y asistencia)

- Este bloque es NO recuperable y su calificación será del 20% de la calificación total en la asignatura.
- La evaluación de este bloque será en función de:
  - Asistencia a clases de laboratorio y tutorías ECTS
  - Resolución de diferentes prácticas periódicas planteadas a través del Aula Virtual.
  - Realización de actividades de autoevaluación
  - Realización de un proyecto de programación concurrente y distribuida. Para superar el proyecto será necesaria la asistencia de, al menos, un 80% de las clases de laboratorio. La calificación dependerá de la actuación y respuesta de todos los miembros del grupo.

La copia o el plagio demostrados en cualquier actividad supone una nota final de SUSPENSO (0) en la convocatoria y una nota de 0 en los bloques no recuperables para todos los implicados, además de las actuaciones legales indicadas según la normativa vigente.

Modalidad Evaluación Final.

Esta modalidad conlleva un examen final único de conocimientos de la asignatura donde podrá alcanzarse un 80% de la nota. No todas las competencias pueden ser evaluadas en este examen final. La evaluación de las destrezas adquiridas en Laboratorio se realiza en pruebas diferentes a la escrita y tradicional en examen oficial. En las prácticas de laboratorio se aborda, entre otras, la CT09, por lo que la asistencia a las mismas es obligatoria. En esas actividades el alumno podría alcanzar un 20% adicional de nota para llegar al 100%.

Bibliografía (básica y complementaria)

Bibliografía Básica

[Ben90] M. Ben-Ari. Principles of concurrent and Distributed Programming. Prentice-Hall

[Bur93] A. Burns, G.L. Davies. Concurrent Programming. Addison-Wesley

[Dav92a] G.L. Davies. Pascal-FC, versión 5. User Guide for Pc compatibles. Universidad de Bradford, UK

[Dav92b] G.L. Davies. Pascal-FC, versión 5. Language Reference Manual. Universidad de Bradford., UK

[Dei98] H.M. Deitel, P.J. Deitel. Cómo programar en Java. Prentice Hall

Fer12] Javier Fernández González. Java Concurrency Cookbook. Editorial Packt Publishing Ltd.

[Fer16] Javier Fernández González. Mastering Concurrency Programming with Java 8. Editorial Packt Publising Ltd

[Gal15] Ricardo Galli. Principios y Algoritmos de Concurrencia

[Har98] S. J. Hartley. Concurrent Programming. The Java Programming Language. Oxford University Press.

[Ray92] M. Raynal. Distributed algorithms and protocols. Ed. John Wiley & Sons.

[Mag99] J. Magee, J. Kramer. Concurrency. State model & Java Programs. John Wiley & Sons

[Oak97] S. Oaks, H. Wong. Java threads. O'Reilly & Associates

[Pal03] J.T. Palma, M.C. Garrido, F. Sánchez, A. Quesada. Programación Concurrente. Editorial Thomson-Paraninfo (www.paraninfo.es)

[Per90] J.E. Pérez. Programación Concurrente. Editorial Rueda.

Bibliografía Complementaria

[And91] G.R. Andrews. Concurrent Programming. Principle and Practice. Addison-Wesley

[Axf89] T. Axford. Concurrent Programming. Fundamental Techniques for Real-Time and Parallel Software Design. Editorial Wiley. Series en Parallel Computing

[Bac98] J. Bacon. Concurrent Systems. Operating Systems, Database and Distributed Systems: An Integrated Approach. Addison Wesley

[Bar98] J. Barnes. Programming in Ada95. Addison-Wesley

[Bev97] J. Beveridge, R. Wiener. Multithreading Applications in Win32. Addison-Wesley

[Bus88] D. Bustard, J. Elder, J. Welsh. Concurrent Program Structures. Prentice Hall International Series in Computer Science

[But97] D. R. Butenhof. Programming with POSIX Threads. Addison-Wesley Professional Computing Series

[Dea00] D. Lea. Programación Concurrente en Java. Principios y patrones de diseño. Addison Wesley

[Dei90] H.M. Deitel. Sistemas Operativos. Addison-Wesley Iberoamericana, 2ª edición

[Geh88] N. Gehani, A.D. McGettrick. Concurrent Programming. International Computer Science Series. Addison-Wesley

[Hol00] Allen Holub. Taming Java Threads. Apress

[Hyd99] Paul Hyde. Java Thread Programming. SAMS (división de MacMillan Computer Publishing)

[Lew00] B. Lewis, D. J. Berg. Multithreaded programming with Java Technology. Sun Microsystems

Press
[Lyn96] Nancy Lynch, Distributed Algorithms, Ed. Morgan Kaufmann Publishers, 1996
[Mil94] M. Milenkovic. Sistemas Operativos. Concepto y diseño. McGraw-Hill, 2ª edición
[Nic96] B. Nichols, D. Buttlar, J. Proulx. Pthreads Programming. O'Reilly

[Pet93] J.L. Peterson, Abraham Silberschatz. Sistemas Operativos. Editorial Reverté

Otros recursos y materiales docentes complementarios

Recursos Virtuales

- Aula virtual de la asignatura
- http://www.eclipse.org
- http://java.sun.com/

Medios materiales utilizados:

- Pizarra
- Cañón de video
- Ordenador
- Internet

Materiales y recursos utilizados:

Los materiales y recursos utilizados estarán disponibles en el espacio reservado para la asignatura en el Campus Virtual. Concretamente los alumnos dispondrán de:

- Trasparencias para cada tema del programa
- Guiones de las sesiones de laboratorio
- Foros de preguntas y respuestas
- Tablón de anuncios de novedades
- Glosarios de términos, palabras claves
- Baterías de preguntas de test
- Conjunto de referencias web relacionadas con la programación concurrente y distribuida
- Tests de autoevaluación de contenidos
- Tareas virtuales para la entrega de problemas propuestos
- Videotutoriales con conceptos teóricos y resolución de ejercicios

Horario de tutorías

Tutorías Programadas: Esta actividad se incluirá en la agenda del estudiante. Se publicitarán en el aula virtual de la asignatura al comienzo del semestre. Aproximadamente se impartirá una hora por alumno durante la sexta semana del semestre y otra hora durante la undécima semana del semestre.

Tutorías de libre acceso: A determinar por el Departamento antes del 31-Julio-2016 y se publicarán en la web del Centro (http://epcc.unex.es). Asimismo, se publicitarán también en el aula virtual de la asignatura y en la puerta del despacho del profesor. Además, la comunicación entre profesor-estudiante y estudiante-estudiante será continua a lo largo de todo el curso mediante el aula virtual.

Recomendaciones

➢ Los estudiantes rellenarán, caso de no haberlo realizado con anterioridad, sus datos en el Aula Virtual (especialmente aportarán una foto reciente y clara) durante las dos primeras semanas de clase.
➢ Es aconsejable la asistencia a clases de teoría, y evaluable la asistencia a las clases de laboratorio.

➢ *Con respecto a la matrícula de la asignatura*:
- **Es altamente recomendable haber superado** las asignaturas de "Introducción a la Programación" y "Desarrollo de Programas".
- Asimismo, **se requiere** el conocimiento de un Entorno de Desarrollo Integrado y del lenguaje de programación **Java**.

➢ *Con respecto al seguimiento de la asignatura durante el curso académico*:
- Para adquirir los conocimientos comprendidos en la asignatura y superarla con éxito es imprescindible consultar de forma habitual el correo electrónico y el Aula Virtual de la materia. En dicho sitio web se mantiene información acerca del tema que se está impartiendo en clase en cada momento así como información acerca de las diferentes actividades prácticas, y de su evaluación.