ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA

GUIA DOCENTE

CURSO 2017/2018

Grado en Ingeniería Informática itinerario Ingeniería del Software

DATOS DE LA ASIGNATURA				
Nombre:				
Métodos Formales en Ingeniería de Software				
Denominación en inglés:				
Formal Methods in Software Engineering				
Código:		**Carácter:**		
606010219		Obligatorio		
Horas:				
	Totales	**Presenciales**	**No presenciales**	
Trabajo estimado:	150	60	90	
Créditos:				
	Grupos reducidos			
Grupos grandes	**Aula estándar**	**Laboratorio**	**Prácticas de campo**	**Aula de informática**
3	0	0	0	3
Departamentos:		**Áreas de Conocimiento:**		
Tecnologías de la Información		Lenguaje y Sistemas Informáticos		
Curso:		**Cuatrimestre:**		
3º - Tercero		Segundo cuatrimestre		

DATOS DE LOS PROFESORES			
Nombre:	**E-Mail:**	**Teléfono:**	**Despacho:**
Roldán Ruiz, Ana María	amroldan@dti.uhu.es	8 7387	51 TU

*Profesor coordinador de la asignatura

Consultar los horarios de la asignatura

DATOS ESPECÍFICOS DE LA ASIGNATURA

1. Descripción de contenidos

1.1. Breve descripción (en castellano):

El objetivo de esta asignatura es que el alumno adquiera competencias en el conocimiento y uso de los lenguajes, técnicas y herramientas más actuales para la construcción de sistemas software de calidad, que le permitirán conocer los aspectos de especificación y análisis de la corrección parcial de sistemas software complejos. Será por tanto necesario adquirir destrezas en :

- Técnicas de descripción formal de sistemas software.
- Análisis de errores mediante comprobación de modelos.
- Interpretación abstracta aplicada a la comprobación de modelos.
- Transformación de programas. Evaluación parcial.
- Aplicaciones. Fiabilidad en lenguajes de programación y en UML.
- Otros métodos para fiabilidad. Demostradores de teoremas. Testing.

1.2. Breve descripción (en inglés):

The main objective of this course is to achieve knowledge and skills in the use of the latest techniques for building reliable software. SLAM, Feaver and Verisoft or JPF are base on these techniques. So, our students must acquire skills in:

- Formal description in analysis of software errors.
- Analysis of errors by model checking.
- Abstract interpretation applied to model checking.
- Transformation of programs. Partial evaluation.
- Applications. Reliability in programming languages and UML.
- Other methods of reliability. Theorem proving. Testing

2. Situación de la asignatura

2.1. Contexto dentro de la titulación:

La ausencia de errores en el software es una constante desde que se diseñaron los primeros lenguajes de programación y su depuración siempre ha sido uno de los mayores problemas de su desarrollo. En los cursos previos a esta asignatura, los alumnos simplemente han hecho uso simplemente de las técnicas de análisis y depuración que proporcionaban los compiladores propios de cada lenguaje de desarrollo. Y será en esta asignatura en la que podrán obtener las destrezas necesarias que les permitan analizar el software complejo de manera completa y correcta.

2.2. Recomendaciones:

Esta asignatura se adentra en la descripción y uso de técnicas que permitan verificar la ausencia de errores en los desarrollos software. Es por ello que el alumno debe tener conocimientos relacionados con dichos desarrollos: conceptos básicos y avanzados de diseño y programación orientada a objetos así como principios y fundamentos de la ingeniería del software y de programación concurrente.

3. Objetivos (Expresados como resultados del aprendizaje):

El objetivo general de la asignatura es proporcionar las herramientas formales que permitan al alumnado:

- Describir y especificar un sistema software concurrente y complejo de forma correcta;
- Eliminar las ambigüedades y depurar sistemas;
- Realizar de forma automática verificaciones de si un sistema satisface su especificación, o detectar casos en los que no, etc..

Todo esto permitirá disminuirá el tiempo dedicado a encontrar errores de diseño y codificación.

4. Competencias a adquirir por los estudiantes

4.1. Competencias específicas:

- **CE3-IS:** Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles
- **CE4-IS:** Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales
- **CE5-IS:** Capacidad de identificar, evaluar y gestionar los riesgos potenciales asociados que pudieran presentarse

4.2. Competencias básicas, generales o transversales:

- **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
- **G01:** Capacidad de organización y planificación así como capacidad de gestión de la Información.
- **G02:** Capacidad de comunicación oral y escrita en el ámbito académico y profesional con especial énfasis, en la redacción de documentación técnica
- **G04:** Capacidad para tomar decisiones basadas en criterios objetivos (datos experimentales, científicos o de simulación disponibles) así como capacidad de argumentar y justificar lógicamente dichas decisiones, sabiendo aceptar otros puntos de vista
- **G05:** Capacidad de trabajo en equipo.
- **G06:** Capacidad para el aprendizaje autónomo así como iniciativa y espíritu emprendedor
- **G08:** Capacidad para adaptarse a las tecnologías y a los futuros entornos actualizando las competencias profesionales.
- **G09:** Capacidad para innovar y generar nuevas ideas.
- **G11:** Respeto a los derechos fundamentales y de igualdad entre hombres y mujeres
- **T01:** Uso y dominio de una segunda lengua
- **T02:** Conocimiento y perfeccionamiento en el ámbito de las TIC’s

5. Actividades Formativas y Metodologías Docentes

5.1. Actividades formativas:

- Sesiones de Teoría sobre los contenidos del Programa.
- Sesiones de Resolución de Problemas.
- Sesiones Prácticas en Laboratorios Especializados o en Aulas de Informática.
- Actividades Académicamente Dirigidas por el Profesorado: seminarios, conferencias, desarrollo de trabajos, debates, tutorías colectivas, actividades de evaluación y autoevaluación.

5.2. Metologías docentes:

- Clase Magistral Participativa.
- Desarrollo de Prácticas en Laboratorios Especializados o Aulas de Informática en grupos reducidos.
- Resolución de Problemas y Ejercicios Prácticos.
- Tutorías Individuales o Colectivas. Interacción directa profesorado-estudiantes.
- Planteamiento, Realización, Tutorización y Presentación de Trabajos.
- Evaluaciones y Exámenes.

5.3. Desarrollo y justificación:

Sesiones Académicas de Teoría y Problemas:
Consisten en clases magistrales paticipativas en las que se impartirán la base teórica de la asignatura y se resolverán ejercicios y problemas aclaratorios de la misma al grupo.
Sesiones Académicamente dirigidas:
Entre otras cosas, en estas sesiones se fomentará la participación del alumnado en tutorías que permitan la interacción con el alumnado, seminarios como complementos a su formación, en la resolución de trabajos estableciéndose como método de entrega la plataforma Moodle fomentando entre otras competencias la innovación, el uso de recursos de e-learning así como el trabajo en equipo.
Prácticas de Laboratorio:
Los alumnos dispondrán con antelación la relación de problemas a resolver y la metodología de trabajo. Las prácticas se realizarán de forma grupal. La asistencia a las sesiones de laboratorio es obligatoria, se pasará lista y no se admitirá más de 1 falta sin justificar por alguna Institución de acreditada solvencia, ya que esta situación conllevará a suspender la parte práctica (laboratorio) y la necesidad de presentarse a un examen final de prácticas para aprobar esta parte de la asignatura.

6. Temario desarrollado:

Bloque I: Estado del arte
1. Introducción a las técnicas de descripción formales en el análisis de errores software
Bloque II: Verificación de propiedades software mediante model checking
2. Fundamentos teóricos del model checking
3. Lenguajes de especificación y herramientas de model checking
Bloque III: Modelado basado en teoría de conjuntos
4. Fundamentos : conjuntos y relaciones
5. Herramientas para la especificación y el análisis
6. SAT-Solvers
Bloque IV: Verificación deductiva de sistemas software
7. Introducción al diseño por contrato
8. Lenguajes de especificación de propiedades
9. Herramientas de verificación de asertos

7. Bibliografía

7.1. Bibliografía básica:

- D. A. Peled, Software Reliability Methods, 2001, Springer.
- E. M. Clarke and O. Grumberg and D. A. Peled, Model Checking, 2000, The MIT Press.
- G.J. Holzmann. The Spin Model Checker, 2008, Addison-Wesley.

7.2. Bibliografía complementaria:

- Ranjit Jhala and Rupak Majumdar. Software model checking. ACM Comput. Surv. 41, 4, Article 21 (October 2009), 54 pages. 2009. DOI=10.1145/1592434.1592438 http://doi.acm.org/10.1145/1592434.1592438
- Clarke E. M., Wing J. M., Formal Methods: State of the Art and Future Directions, ACM Workshop on Strategic Directions in Computing Research, ACM Computing Surveys vol. 28(4): 626-643, 1996.
- Gunter C., Mitchell J., Strategic Directions in Software Engineering and Programming Languages, ACM Workshop on Strategic Directions in Computing Research, ACM Computing Surveys, 28(4).

8. Sistemas y criterios de evaluación.

8.1. Sistemas de evaluación:

- Examen de teoría/problemas
- Defensa de Prácticas
- Defensa de Trabajos e Informes Escritos
- Examen de prácticas

8.2. Criterios de evaluación y calificación:

1.- Conocimientos teóricos: Examen teórico escrito al final del cuatrimestre (ver calendario fijado por la ETSI) con un valor del 50% de la calificación final de la asignatura.
2.- Conocimientos prácticos de laboratorio: Conjunto de prácticas que representan el 40% de la calificación final de la asignatura.
3.-Actividades académicamente dirigidas: Ejercicios/trabajos entregables expuestos por los alumnos y/o conferencias/tutorías colectivas que suponen el 10% de la calificación final de la asignatura.
Nota en Acta = Examen Teórico Escrito+ Actividades Prácticas Laboratorio +Actividades académicamente dirigidas.
NOTA ADICIONAL:
A.- La asignatura se considera aprobada si se obtiene como Nota en Acta un valor mayor o igual a 5 (sobre 10).

Se garantiza la adquisición de las competencias de la siguiente forma: mediante la evaluación del examen teórico, las competencias CE4-IS, G01, G06, G08 y T02; mediante los conocimientos prácticos de laboratorio, las competencias CE3-IS, CE4-IS, CB3, G04, G05, G06, G08, G09, G11 y T02 y mediante las actividades académicamente dirigidas, las competencias CE4-IS, CE5-IS, CB3, G02, G05, G08, G09, G11 y T02.

9. Organización docente semanal orientativa:

Semanas	Grupos Grandes	Grupos Reducidos Aula Estándar	Grupos Reducidos Aula de Informática	Grupos Reducidos Laboratorio	Grupos Reducidos prácticas de campo	Pruebas y/o actividades evaluables	Contenido desarrollado
#1	2	0	2	0	0		Presentación/Bloque I
#2	2	0	2	0	0		Bloque II
#3	2	0	2	0	0		Bloque II
#4	2	0	2	0	0		Bloque II
#5	2	0	2	0	0		Bloque II
#6	2	0	2	0	0		Bloque II
#7	2	0	2	0	0		Bloque II
#8	2	0	2	0	0		Bloque III
#9	2	0	2	0	0		Bloque III
#10	2	0	2	0	0		Bloque III
#11	2	0	2	0	0		Bloque III
#12	2	0	2	0	0		Bloque IV
#13	2	0	2	0	0		Bloque IV
#14	2	0	2	0	0		Bloque IV
#15	2	0	2	0	0		Bloque IV
	30	0	30	0	0		