

# Diseño Software

Pablo Sánchez

Dpto. Ingeniería Informática y Electrónica  
Universidad de Cantabria  
Santander (Cantabria, España)  
p.sanchez@unican.es



# Índice

- 1 **Introducción**
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Profesorado

Pablo Sánchez Barreiro

Despacho 1069

Departamento de Ingeniería Informática y Electrónica  
p.sanchez@unican.es

Diego García Saiz

Despacho 1071

Departamento de Ingeniería Informática y Electrónica  
diego.garcia@unican.es

# Horario Clases

	Lunes	Martes	Miércoles	Jueves	Viernes
08:30 - 09:30					
09:30 - 10:30					
10:45 - 11:45		Diseño	Diseño	Diseño	
11:45 - 12:45		Diseño			
12:45 - 13:45					

- 1 Para acudir a tutorías con los profesores se podrá hacer a cualquier hora, preferentemente en horario de mañana.
- 2 Para asegurar la disponibilidad y atención al alumno, se recomienda avisar con antelación.

# Índice

- 1 Introducción
- 2 **Objetivos y Temario**
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Índice

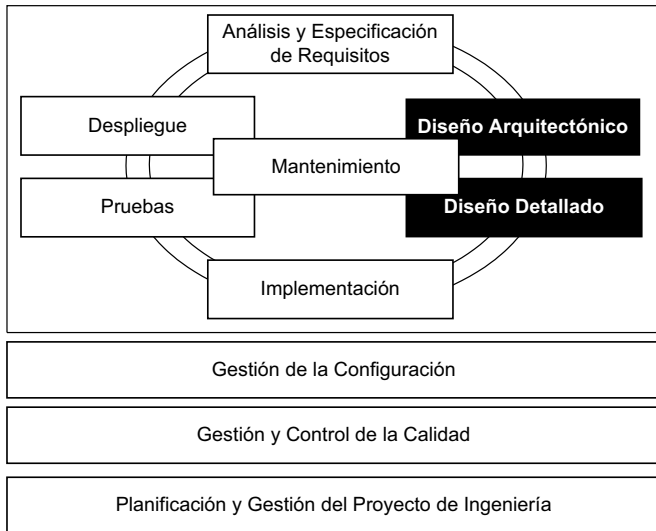
- 1 Introducción
- 2 Objetivos y Temario
  - **Objetivos**
  - Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Objetivos de la Asignatura

## Objetivos

Saber utilizar los principios, fundamentos, métodos, técnicas y herramientas para poder abordar satisfactoriamente las fases de **diseño arquitectónico** y **diseño detallado** de un proyecto software de mediana escala y perteneciente principalmente al dominio de los sistemas empresariales.

# Objetivos de la Asignatura





# Objetivos de Aprendizaje

- 1 Saber aplicar los principios GRASP y SOLID.
- 2 Saber aplicar los principios de Diseño por Contrato.
- 3 Conocer y comprender la relación existente entre *patrón de diseño*, *antipatrón* y *refactorización*.
- 4 Saber aplicar patrones de diseño microarquitectónicos, en especial, los patrones GoF.
- 5 Conocer y comprender algunos de los tipos principales de *antipatrones* existentes.
- 6 Conocer y comprender algunos de los tipos principales de *refactorizaciones* existentes.
- 7 Conocer y comprender el concepto de arquitectura sw.
- 8 Saber aplicar patrones de diseño arquitectónicos, en especial los relativos a *arquitecturas empresariales en tres capas*.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
  - Objetivos
  - **Temario**
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Temario

- ➊ Principios Fundamentales del Diseño Software.
- ➋ Patrones de Diseño Software.
- ➌ Diseño e Implementación de Arquitecturas Software.
- ➍ Patrones Arquitectónicos para Arquitecturas Empresariales.
- ➎ Medición de las Propiedades Fundamentales de un Diseño Software.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
  - **Plataforma**
  - Actividades
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Plataforma de Trabajo

- La plataforma de trabajo será la asignatura es *moodle*.
- Todas las notificaciones, publicaciones y entregas se harán a través de *moodle*.
- Es obligación del alumno estar atento a las posibles notificaciones y avisos que se realicen a través de moodle.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
  - Plataforma
  - **Actividades**
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Clases de Aula

## Objetivo

Entender los conocimientos teóricos que constituyen la base de las habilidades y destrezas que se desarrollarán a lo largo de la asignatura.

- Sin conocimiento teórico es imposible alcanzar las habilidades prácticas.
- La asistencia a las clases teóricas y prácticas no es obligatoria, **pero si altamente recomendable e incluso necesaria**.
- La asignatura no está diseñada para ser seguida a distancia.
- Las clases serán fundamentalmente lecciones magistrales combinadas con alguna metodología activa.
- **Las transparencias**, cuando las haya, **no serán apuntes**.
- Todo tema tendrá asociado un itinerario para su aprendizaje de forma autónoma.



# Clases de Laboratorio

## Objetivo

Aplicar los conceptos teóricos aprendidos en las clases de aula al desarrollo de un sistema software real de mediana escala, con el objetivo de desarrollar las competencias procedimentales y actitudinales deseadas.

- Desarrollo de pequeñas prácticas individuales sobre problemas concretos.
- Se implementarán patrones de diseño en C# y Java.
- No hay una receta única para hacer las prácticas, existen diferentes soluciones alternativas posibles.
- Lo importante no es sólo que las prácticas funcionen, sino que los patrones que tratan de ejercitar se hayan aplicado correctamente.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 **Métodos de Evaluación/Calificación**
- 5 Bibliografía

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
  - **Fórmula Calificación**
  - Pruebas Evaluables
  - Evaluación de las Prácticas
- 5 Bibliografía

# Cálculo de la Calificación Final

## Fórmula de Cálculo de la Calificación Final

$$\begin{aligned}\text{Calificacion Final} = & \text{PruebaPatrones} \quad \times 0.50 + \\ & \text{PruebaArquitectura} \quad \times 0.50 + \\ & \text{TorreBabel} \quad \times 0.10\end{aligned}$$

# Aclaraciones sobre la Evaluación

- La prueba de patrones se celebrará a finales de Noviembre o principios de Diciembre.
- La prueba de arquitectura se celebrará en la convocatoria de exámenes de Febrero.
- Una vez superada cualquiera de las pruebas, el alumno no tendrá que volver a realizarla dentro de ese curso académico.
- La prueba de patrones es recuperable en la convocatoria de Febrero.
- Un alumno puede repetir una prueba si desea mejorar su calificación, pero en ese caso sólo computará la calificación de la última prueba realizada.
- Una calificación media de 4.99 es suspenso.
- La prueba opcional de la Torre de Babel puede servir para aprobar si se superan unos mínimos.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
  - Fórmula Calificación
  - Pruebas Evaluables
  - Evaluación de las Prácticas
- 5 Bibliografía

# Pruebas Evaluables

- Las pruebas consistirán en pruebas prácticas de laboratorio que podrán contener además razonamientos sobre cuestiones teóricas.
- Se puede hacer uso en las pruebas de todo el material escrito o digital que se desee.
- El material escrito debe servir para consultar cuestiones puntuales, pero en el caso ideal no debería hacerse ningún uso del mismo.
- Se pueden preguntar durante las pruebas cuestiones relativas a la sintaxis de los lenguajes de programación o la utilización de sus librerías.
- Queda prohibido durante las pruebas el uso de dispositivos con capacidad de comunicación inalámbrica.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
  - Fórmula Calificación
  - Pruebas Evaluables
  - Evaluación de las Prácticas
- 5 Bibliografía



# Evaluación de las Prácticas

- No se realizarán entregas de prácticas semanales durante el curso.
- Como parte de las pruebas de laboratorio se podrá solicitar la entrega de una práctica o la realización de varias modificaciones sobre algunas de ellas.
- Las prácticas se pueden hacer en grupo, siempre y cuando se reconozca la autoría grupal.
- Se puede solicitar al profesor, y se recomienda hacerlo, una evaluación superficial de las prácticas antes de la realización de las pruebas evaluables.

# Índice

- 1 Introducción
- 2 Objetivos y Temario
- 3 Metodología
- 4 Métodos de Evaluación/Calificación
- 5 Bibliografía

# Bibliografía Principal



Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides.  
*Design Patterns: Elements of Reusable Object-Oriented Software.*  
Addison Wesley, November 1994.



Martin Fowler.  
*Patterns of Enterprise Application Architecture.*  
Addison-Wesley Professional, 2002.

# Bibliografía Secundaria



Craig Larman.

*Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development.*

Prentice Hall, 3 edition, OctoberPrentice 2004.



Robert C Martin.

*Agile Software Development, Principles, Patterns, and Practices.*

2002.



Bertrand Meyer.

*Touch of Class: Learning to Program Well with Objects and Contracts.*

Springer, September 2009.

# Bibliografía Secundaria



Erick Freeman, Elisabeth Robson, Bert Bate, and Kathy Sierra.  
*Head First Design Patterns*.  
O'Reilly, 2004.



William J. Brown, Raphael C. Malveau, Hays W. “Skip” McCormick,  
and Thomas J. Mowbray,  
*AntiPatterns: Refactoring Software, Architectures and Projects in  
Crisis*  
Wiley, 1998.