# Desarrollo de Software Basado en Componentes y Servicios (DSBCS)

M.I. Capel

Departamento de Lenguajes y Sistemas Informáticos
Email: manuelcapel@ugr.es
http://lsi.ugr.es/mcapel/

9 de octubre de 2020

# Presentación de la asignatura

## Información General

- Transmisión de las clases ”on line” en Google Meet: https://meet.google.com/hya-kuud-gdg
- Materiales didácticos en: en https://pradoposgrado2021.ugr.es/ ->“Desarrollo de Sistemas de Software basados... -M502_56_8_2021”

## Sobre el profesor

- Profesor: Manuel I. Capel https://lsi.ugr.es/mcapel/ email: `manuelcapel@ugr.es`
- Despacho: A-37 ($3^a$ planta)

## Tutorías

| L | M | X | J | V |
|---|---|---|---|---|
| 11.30-13.30 | 9.30-11.30 | 15.30-17.30 | - | - |

Cuadro: (*) se puede acordar una tutoría con el profesor fuera de este horario enviando un mensaje a manuelcapel@ugr.es

# Coordenadas de la asignatura

## DDBCS

Máster en Ingeniería Informática
Máster Universitario en Ingeniería Informática
Módulo: Tecnologías Informáticas 1
Materia: Sistemas Basados en Componentes y Servicios
Carácter: Obligatoria
Carga docente: 4 cr.
Grupos teoría: 1 (viernes 15.30-17.00, 1.6)
Grupos prácticas: 2 (Jueves DSS2:17.00-18.30, ?; Jueves DSS1:18.30-20.00, ?)

## DDBCS

Teoría: 12 sesiones= 18,0 hrs (comienza: 9/10/2020; termina: 15/01/2021)
Prácticas: 12 sesiones= 18,0 hrs (comienza: 9/10/2020; termina: 21/01/2021)
+ 4,5 horas dedicadas a la defensa/exposición (22, 28 y 29/01/2021) de trabajos y sesión de evaluación

# Objetivos formativos

## Objetivos (BOE 8 de junio de 2009 )

- Comprender los modelos de componentes actuales
- Tendencias de desarrollo de software con énfasis en componentes y servicios
- Apreciar lo que reporta basarse en componentes y servicios al desarrollar software, espec. en ”validación”
- Modelos formales esenciales de respaldo
- Arquitecturas software actuales, patrones y estilos, así como conocer su impacto en el desarrollo de software
- Planificar la evolución de un sistema software y evaluar el nivel de calidad que mantiene
- Fundamentos, herramientas y distribuciones libres disponibles del software de intermediación
- Arquitecturas de servicios y cómo aplicar las metodologías y tecnologías apropiadas en casos de aplicación

# Objetivos formativos II

## Capacidades que se han de adquirir

- Aplicar las arquitecturas más adecuadas, los componentes que las integran, las interfaces que se definen entre ellos, los patrones que supervisan su composición
- Diferenciar los paradigmas “Grid Computing” y “Cloud Computing”, sabiendo cuál aplicar en cada caso
- Obtener provecho de conductores de software inspirados en software intermediario para desarrollar aplicaciones y servicios específicos
- Diseñar y utilizar marcos de trabajo para la construcción de sistemas software distribuidos de calidad en diferentes dominios de aplicaciones
- Más información en: http://masteres.ugr.es/ing-informatica

# Programa de Teoría

I. Desarrollo de software basado en componentes y servicios
   - Formalización de los sistemas abiertos y basados en componentes.
   - Técnicas de diseño y desarrollo basadas en componentización del software
II. Servicios Web
   - Limitaciones del software intermediario (middleware) convencional.
   - Middleware y arquitecturas de servicios
   - Servicios Web contemporáneos (WS 2.0)
   - Notaciones y lenguajes
   - Programación de SW
III. Modelado de procesos de negocio
   - Desarrollo de software, basado en SW, para procesos de negocio
   - Composición de SW: orquestación y coreografía.
   - Notaciones de modelado actuales
IV. Sistemas Ubicuos e Inteligencia Ambiental
   - Introducción a la Computación Ubicua
   - Marcos de trabajo actuales de desarrollo
   - Servicios colaborativos

**Nota**

Todos los materiales didáticos de la asignatura, excepto algún tutorial sobre librerías y lenguajes, están escritos en inglés, así como todas las diapositivas de las clases de teoría. Se recomienda un libro de texto en español para seguir el curso.

# Seminarios (prácticos)

## Tema

I Especificación de componentes software con UML/OCL

II Introducción al diseño/implementación/despliegue de servicios Web

III Introducción a la orquestación de servicios Web complejos: WS-BPEL

IV OWL y modelado semántico de ontologías para la Web Semántica

Los seminarios podrán incluir la entrega de ejercicios prácticos y/o programas propuestos por el profesor

## Nota

Se realizará un cuestionario de forma individualizada y con control de tiempo (50') y de disponiblidad (se avisará la fecha y hora) para hacerlo, al final de cada tema de teoría.

# Programa de Prácticas

1. Programación de componentes-software distribuidos
2. Desarrollo de un servicio Web con persistencia de entidades e interfaz REST
3. Modelado de procesos de negocio propuestos con BPEL 2.0
4. Desarrollo completo de una aplicación receptiva y adaptable para dispositivos móviles y su interfaz Web; programación completa del servicio y base de datos en la parte servidora

| Práctica(*) | Fecha indicativa de comienzo |
|---|---|
| 1 | 15/Octubre/2020 |
| 2 | 22/Octubre/2020 |
| 3 | 5/Noviembre/2020 |
| 4 | 26/Noviembre/2020(*) |

12 sesiones de prácticas + 1 sesión de defensa/exposición

(*) Fecha límite de entrega de prácticas: 21/01/2021 (23:00 hrs)

# Modelo de evaluación: *sistema de evaluación continua*

## Teórico (NET)

- varias pruebas de comprensión (*tests*) y entregas de ejercicios (*tareas*), sobre el desarrollo y los resultados, todo se realizará en ”Prado”, en las fechas establecidas para cada una de las citadas pruebas y entregas

## Práctico (NEP)

- asistencia a prácticas obligatoria (máximo 2 faltas justificadas) para ser calificado con el *sistema de evaluación continua*
- se pueden realizar de forma individual (prácticas 1-3) o la práctica 4 (solamente) en grupos de 2, como máximo;
- entrega de resultados parciales, que serán calificados, antes de la entrega final de la práctica 4 (del 21/01/2021)
- reuniones de seguimiento de las prácticas con el profesor
- la última práctica realizada se expondrá (por todos los participantes) a toda la clase, en fecha y hora anunciada con anterioridad

# Modelo de evaluación: *evaluación única* y convocatoria de septiembre

- **Evaluación:** examen teórico y práctico (10 puntos)
- **Evaluación en la convocatoria de septiembre para ambas modalidades:** Sólo se realizará en esta convocatoria la evaluación con un único examen teórico/práctico. La evaluación de las prácticas entregadas antes del 3/09/2021 será el 50 % de la calificación final.

# Documentación de la asignatura

Toda la documentación de la asignatura se gestiona en la plataforma Prado (https://pradoposgrado2021.ugr.es/course/view.php?id=2480 ), así como:

- Notificación de resultados de pruebas y exámenes
- Entrega de ejercicios, trabajos y prácticas
- Comunicación entre alumno y profesor

# Recomendaciones

## Es importante:

- Utilizar las horas de tutoría
- Mantenerse informado de la marcha de la asignatura: acceder al sitio de la asignatura *Desarrollo de Sistemas de Software basados en Componentes y Servicios-1920*) a través de acceso identificado en *Prado* https://pradoposgrado1920.ugr.es/course/view.php?id=13238
- Participar en clase
- Revisar la bibliografía, no limitarse sólo a leer las diapositivas

# Bibliografía:



Aalst, W. v. d., Benatallah, B., Casati, F., Curbera, F., and Verbeek, H. (2007).
Business process management: Where business processes and web services meet (guest editorial).
*Data & Knowledge Engineering*, 61(1):1–5.


Armbrust, M. and etal. (2009).
*Above the Clouds: A Berkeley View of Cloud Computing*.
ACM.


Armbrust, M. and etal. (2010).
A view of cloud computing.
*Communications of the ACM*, 53:50–58.


Bell, M. (2010).
*SOA Modeling Patterns for Service Oriented Discovery Analysis*.
Wiley, Complementaria.


Capel, M. (2016).
*Desarrollo de software ys sistemas basados en componentes y servicios*.
Garceta, **Básica**.


Clements, P., Bachmann, F., Bass, L., Garlan, D., Ivers, J., Little, R., Merson, P., Nord, R., and Stafford, J. (2010).
*Documenting Software Architectures: Views and Beyond*.
Addison-Wesley, second edition.


deSilva, L. and Balasubramaniam, D. (2012).
Controlling software architecture erosion: a survey.
*Journal of Systems and Software*, 85(01/2012):132–151.


Developer and Works (2009).
*Cloud computing versus grid computing*.
IBM.


Erl, T. (2004).
*Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services*.
Prentice-Hall.


Erl, T. (2008).
*SOA. Principles of Service Design*.
Prentice-Hall.


Fowler, M. (2003).
*Patterns of enterprise application architecture*.
Addison–Wesley, Boston, Mass.


Lungu, M. (2008).
*Software architecture recovery*.
University of Lugano, http://www.slideshare.net/mircea.lungu/software-architecture-recovery-in-five-questions-presentation, 2008 edition.


Maranzano, J., Rozsypal, S., Zimmerman, G., Warnken, G., Wirth, P., and Weiss, D. (2005).
Architecture reviews: Practice and experience.
*IEEE Software*, 22(2).


**Marcs, E.D. and Bell, M.** (2006).
*Service Oriented Architecture (SOA): A Planning and Implementation Guide for Business and Technology*.
Wiley, **Básica**.


Osterwalder and Pigneur (2004).
An ontology for e-business models.
*Butterworth-Heinemann*, pages 65–97.


Silver, B. (2011).
*BPMN Method and Style: with BPMN Implementer's Guide*.
Cody-Cassidy Press, Complementaria.


**Szyperski, C.** (1998).
*Component Software. Beyond Object-Oriented Programming*.
Addison–Wesley.


Tang, A., Han, J., and Vasa, R. (2009).
Software architecture design reasoning: A case for improved methodology support.
*IEEE Software*, 26(2).


Taylor, H. (2009).
*Event-driven Architecture: How SOA Enables the Real-time Enterprise*.
Addison–Wesley, Complementaria.


Terra, R., Valente, M., Czarnecki, K., and Bigonha, R. (2012).
Recommending refactorings to reverse software architecture erosion.
In *16th European Conference on Software Maintenance and Reengineering*.


Verissimo, P. and Rodrigues, L. (2004).
*Distributed Systems for System Architects*.
Kluwer Academic.


von der Beeck, M. (2000).
Behaviour specifications: Equivalence and refinement notions.
In *Visuelle Verhaltensmodellierung Verteilter und Nebenlaeufiger Software–Systeme, 8. Workshop des Arbeitskreises GROOM der GI Fachgruppe 2.1.9 OO Software-Entwicklung*, pages 1–5, Paderborn, D. Universitaet Munster.


Woods, E. (2012).
Industrial architectural assessment using tara.
*Journal of Systems and Software*, 85(9):2034–2047.


zur Muehlen, M., Nickerson, J. V., and Swenson, K. D. (2005).
Developing web services choreography standards—the case of REST vs. SOAP.
*Decission Support Systems*, 40(1):9–29.

# Texto


Garceta
ees
ects
EEES
ECTS
Desarrollo de
software y sistemas
basados en
componentes y servicios
Manuel I. Capel Tuñón

# Biblio complementaria